Research Note 86-10

VIDEODISC INTERPERSONAL SKILLS TRAINING AND ASSESSMENT (VISTA): SOFTWARE AND EVALUATION DETAILS, VOLUME 4

James E. Schroeder and Frederick N. Dyer
Army Research Institute

Paul Czerny, Edward W. Youngling, and Daniel P. Gillotti
Mellonics Systems Development Division, Litton Systems Incorporated

ARI FORT BENNING FIELD UNIT
Joel D. Schendel, Acting Chief

TRAINING RESEARCH LABORATORY
Seward Smith, Acting Director

U. S. Army

Research Institute for the Behavioral and Social Sciences

January 1986

U. S. ARMY RESEARCH INSTITUTE
FOR THE BEHAVIORAL AND SOCIAL SCIENCES

A Field Operating Agency under the Jurisdiction of the

Deputy Chief of Staff for Personnel

EDGAR M. JOHNSON
Technical Director

WM. DARRYL HENDERSON
COL, IN
Commanding

Research accomplished under contract for
the Department of the Army

Mellonics Systems Development Division, Litton Systems Incorporated

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: ARI Research Note 86-10	2. GOVT ACCESSION NO.: ADA166867
3. RECIPIENT'S CATALOG NUMBER:	
4. TITLE *(and Subtitle)*: VIDEODISC INTERPERSONAL SKILLS TRAINING AND ASSESSMENT (VISTA): SOFTWARE AND EVALUATION DETAILS, VOLUME 4	5. TYPE OF REPORT & PERIOD COVERED: Final Report August 1980 - March 1983
	6. PERFORMING ORG. REPORT NUMBER:
7. AUTHOR(s): James E. Schroeder, Frederick N. Dyer (ARI), Paul Czerny, Edward W. Youngling, and Daniel P. Gillotti (Litton/Mellonics)	8. CONTRACT OR GRANT NUMBER(s): MDA 903-80-C-0545
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Mellonics Systems Development Division, Litton Systems, Inc., P.O. Box 2498 Fort Benning, Georgia 31905	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 2Q263743A794
11. CONTROLLING OFFICE NAME AND ADDRESS: U.S. Army Research Institute for the Behavioral and Social Sciences - Fort Benning Field Unit P.O. Box 2086, Fort Benning, Georgia 31905	12. REPORT DATE: January 1986
	13. NUMBER OF PAGES: 156
14. MONITORING AGENCY NAME & ADDRESS *(if different from Controlling Office)*: U.S. Army Research Institute for the Behavioral and Social Sciences. 5001 Eisenhower Avenue Alexandria, VA 22333-5600	15. SECURITY CLASS. *(of this report)*: Unclassified
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE:

16. DISTRIBUTION STATEMENT *(of this Report)*

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

18. SUPPLEMENTARY NOTES

The Contracting Officer's Representative was Seward Smith. The VISTA report has been published in 4 volumes; other volumes published as follows: Volume 1, ARI Technical Report 703; Volume 2, appendixes A - D in ARI Research Note 86-08, and Volume 3, appendixes E - J in ARI Research Note 86-09.

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

computer-assisted instruction
leadership
counseling
training development
interpersonal skills
interactive training
videodisc

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

The Videodisc Interpersonal Skills Training and Assessment (VISTA) project was initiated as a means to use computer-assisted training/videodisc technology to reduce the high training costs associated with junior officer leadership skills training. Historically the major problem was simulating subordinates as they would probably respond in a given leadership situation; assessment center simulations and role playing could train leadership skills but not without high personnel costs due to the numbers of counselors and role players required.

(continued)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

item 20. Abstract (continuation)

Previous research indicated that a videodisc system could successfully train soldier skills even when only a fraction of the capabilities of the medium were used. Such a system could be used to supplement the current role playing and, hence, reduce the number of support personnel required.

The research effort included topic analysis, hardware selection, software development, scenario writing, studio production, editing, and videodisc mastering. Final evaluation of the videodiscs produced included the administration of two tests, a test designed to measure the acquisition of leadership skills and a subjective preference test designed to measure user acceptance.

Nine highly interactive videodisc training scenarios covering 20 leadership problems were produced. Overall evaluation results indicated a VISTA superiority followed by role playing and programmed text, with the majority of students indicating that a combination of videodisc and role playing would be optimal for leadership training. Results also indicate that although VISTA products were designed for the Infantry Officer's Basic Course, the problems addressed are probably common to other Army branches and should therefore be investigated for possible application in other training centers. Key words:

VIDEODISC INTERPERSONAL SKILLS TRAINING AND ASSESSMENT (VISTA)

EXECUTIVE SUMMARY

Introduction:

The U.S. Army's VISTA (Videodisc Interpersonal Skills Training and Assessment) project was initiated to determine whether leadership and counseling skills could be trained using current computer-assisted instruction/. videodisc technology. The target audience was Army junior officers (Second Lieutenants in the Infantry Officer's Basic Course at Fort Benning, Georgia). Five government agencies and two contractors were involved in this effort that included a front-end topic analysis, hardware selection, software development, scenario writing, studio production, editing, videodisc mastering, and final evaluation. The final evaluation compared the VISTA products with a programmed text containing the same information and role playing using the same topic themes. All seven VISTA videodiscs (nine scenarios) were tested.

Procedure:

The first stage of the project involved a front-end topic analysis, hardware and software selection, and design of the instruction. In the topic analysis, 57 candidate interpersonal problem situation topics were generated and rated by 58 subject matter experts for difficulty, importance, and frequency. Situations involving the highest composite scores for the three dimensions were subsequently addressed in the training scenarios. Twenty problem situations were covered in the 9 scenarios produced to date. The hardware system selected comprised an Apple 2+ computer, a DiscoVision videodisc player, a Sony monitor and other assorted peripherals. The software language chosen was Pascal. Two instructional modes of presentation were designed. The Experiential mode simulates a roleplaying situation. There is no textual feedback and the students can go several steps off the "best path". In the Pedagogical mode, extensive textual feedback is presented and the student is never allowed to go more than one step off the best path.

The second major stage of the project involved the scenario writing and the software development. A scenario authoring aid was developed. Guidance for the determination of appropriate alternatives was derived from the two U.S. Army field manuals dealing with leadership and counseling, subject matter experts, and various theoretical approaches for counseling and leadership. The software was developed to complement the instructional design. In addition, software was developed to allow relatively simple entry of textual information and videodisc frame numbers to expedite future videodisc development efforts.

The final stage involved the evaluation of the seven videodiscs (9 scenarios). An experimental evaluation conducted on all training products measured both learning of leadership principles and the student's acceptance of the new instructional technology.

Findings:

The overall results of the evaluation indicated a VISTA superiority followed by role playing and programmed text, respectively, on a test designed to measure the acquisition of leadership principles. Also, both role playing and videodisc were rated high on a subjective preference scale used to measure user acceptance. Role playing was slightly but significantly higher than videodisc and both videodisc and role playing were much higher than the programmed text. The great majority of the students indicated that a combination of videodisc and role playing would be optimal for leadership training.

Products completed:

- Nine scenarios which address 20 problem areas have been produced and evaluated. Overall results indicated a significant superiority of VISTA products over both role playing and programmed text.

- Two instructional approaches designed to optimize the training impact of the new technology.

- Scenario authoring workbook to aid future scenario writing.

- Generic software that will control any of the videodiscs developed played by either of two popular videodisc players, with or without maintenance of detailed student records, and with choice of two instructional modes.

Utilization:

- The VISTA products were implemented in the Counseling Laboratory of the IOBC in June, 1983.

The following is recommended:

- Due to the success of the VISTA project and other videodisc training projects, the U.S. Army should continue to investigate other possible areas for application of computer-assisted instruction/videodisc training.

- Develop standards in both hardware and courseware structure.

- The VISTA products should serve as a supplement to current leadership training approaches rather than a replacement of those approaches.

- The IOBC Counseling Laboratory is currently taught in two periods, one at the beginning of IOBC and one toward the end. Because of the standardized format, role playing should be conducted in the second laboratory as a performance test and the VISTA products should be utilized in the early laboratory (while students are at an early stage in their learning of leadership).

- Although the VISTA products were developed for the Infantry Officer's Basic Course, the problems addressed are probably common to the other branches. Therefore, the VISTA products should be investigated for possible application in other training centers.

ACKNOWLEDGEMENTS

This study was originated by Dr. Frederick N. Dyer at the U.S. Army Research Institute, Fort Benning Field Unit. A total of five government agencies and two contractors were eventually involved. All seven organizations contributed to the success of the project.

Litton Mellonics was the primary contractor responsible for the great majority of the work effort. The original team was headed by Dr. James E. Schroeder who coordinated the overall effort and designed the instruction and evaluation. Dr. Paul Czerny was responsible for the hardware and software selection and the software development. Mr. Daniel P. Gillotti was the Leadership/Counseling subject matter expert responsible for the development of the scenario content. Dr. Edward W. Youngling was the Program Manager of the Litton Mellonics effort for the entire duration of the contract. Over the months, a number of other Litton employees were involved and made significant contributions: (alphabetically) Dr. Gary C. Bayer; Mr. W. Alfred Cook, Jr; Mr. Harry A. Lucker; Dr. Mary N. Perkins; Dr. Mike S. Perkins; Dr. Robert Pleban; Mr. David W. Reiss; and Dr. Gary P. Williams.

The U.S. Army Research Institute, Fort Benning Field Unit supervised the research effort. Special acknowledgement is exended to COL Franklin A. Hart, COL L. Neale Cosby, Dr. Frederick N. Dyer, Dr. Seward Smith, and Mr. Hal Strasel who all provided excellent management, guidance, and suggestions. In addition, Dr. John C. Morey and Sid Hall (an Auburn University doctoral candidate working with ARI through and the Cooperative Education Program at Auburn University), both provided valuable assistance in the data analysis. Also, thanks to MAJ Charles J. Slimowicz, the Research Coordinator at ARI, Fort Benning for his valuable input and for his assistance in securing troop support.

A special acknowledgement is given to the many individuals who volunteered to serve as actors for the six programs. For the most part, these were active duty soldiers who voluntarily arranged their own work schedules to accommodate the VISTA production schedule. Also, some of the actors were volunteers from Litton Mellonics and ARI at Fort Benning.

Fort Benning's Training Audiovisual Support Center (TASC) provided the facilities and expertise for the production of five of the six programs and editing of all six programs. Special credit is extended to Mr. Rubin Webster, Mr. Randy Amos, and Mr. Bennett Yeilding and their staff. The TASC at Fort Gordon provided the actors, facilities, and expertise for the production of the "Performance Counseling" program. Special credit is given to MAJ Doug Dooley, Mr. Gaylord Cavallaro and their staff.

The Training Development Institute (TDI) at Fort Monroe, VA provided funds for the topic analysis, instructional design, software development, scenario development, and evaluation. Acknowledgement is given to COL F. A. Nerone, COL Edmund J. Glabus, Ms. Janet Lamb, Ms. Jean Rose, Mr. Donald A. Kimberlin, and Mr. Frank E. Giunti for their valuable guidance and comments.

The Army Communicative Technology Field Office (ACTO) provided the hardware for the development as well as the funds for the videodisc mastering which was completed by Discovision (later Pioneer Video). Special thanks are extended to COL John A. Goetz, Mr. Bob Reynolds, Mr. Pete Benden, and CPT John Thompson from ACTO for their valuable coordination and assistance.

The U.S. Army Infantry School has made great contributions to the success of the project. Appreciation is extended to all the departments involved. Special thanks go to COL William L. Shackelford and Mr. Walter G. Gardner from the Directorate of Training and to the entire Leadership Department staff who, over the two year period, provided excellent suggestions and guidance (especially): LTC Richard G. Stillwell; MAJ Burton G. Lockwood, II; MAJ Donald E. Allison; MAJ Carl E. Linke; MAJ Edward L. Williams; MAJ Carl B. Fedde; MAJ Larry L. Owens; CPT Theodore Wiggins, Jr.; CPT Craig F. Bennedict; CPT Willard I. Ghere; and CPT Charles L. Smith).

APPENDIX K

SOFTWARE DETAILS

Paul Czerny, Ph.D.

W. Alfred Cook, Jr.

Mellonics Systems Development Division, Litton Systems, Inc.

APPENDIX K

SOFTWARE DETAILS

Table of Contents

Part		Page
1.	Functional Software Overview	K-1
2.	Instructional Users Guide.	K-7
3.	Additional Support Software.	K-8
4.	Lesson Administrative Maintenance Program.	K-10
5.	Student File Format.	K-25
6.	Data Entry Overview.	K-27
7.	Instructions For Running Videodisc	K-30
8.	Software Listings.	K-31
	Videodisc.	K-31
	LAMP	K-49
	Reset.	K-93
	Look	K-96
	Align.	K-101
	Unit Work.	K-104

APPENDIX K

Part 1

Functional Software Overview

The following discussion assumes that the reader has already read and understood the section titled Software Description in the first section of this paper. The videodisc software, as a total system, consists of two separate programs: Videodisc and LAMP (Lesson Administrative Maintenance Program). Videodisc is the program that executes the videodisc lesson. LAMP is used to load the information text files for a specific videodisc lesson.

Videodisc

A brief description of the files used by Videodisc will be given followed by a discussion of the functioning of Videodisc itself. Videodisc relies on the information stored in five files for its operation. These files are INTRO, TEXT.ONE, TEXT.TWO, POINT.ONE, AND STUDENT. INTRO contains the text of the overview that is presented to the student at the beginning of the lesson. This overview describes how the program functions in both the pedagogical and experiential modes. TEXT.ONE contains the text that is displayed as feedback in the pedagogical mode. POINT.ONE contains information as to where (in a particular file) text is located, how many pages of text should be accessed, the start and stop frame numbers for the motion sequences, the number of pages contained in file INTRO, and the start and stop frame numbers for the lightpen or touch-panel instructions. STUDENT contains the identification numbers of the student files that have been used.

The following is a discussion of the operation of Videodisc at a general level. For a more detailed understanding of how the program works, refer to the section labeled Information contained within POINT.ONE. Figure 8 is a simplified diagram of the data flow into and out of Videodisc. When appropriate, text is brought in from one of the text files for display on the monitor. This text might be the introduction to the videodisc lesson (stored in INTRO), text prior to and/or following the first motion sequence (stored in TEXT.TWO). The main information file, POINT.ONE, dictates how the program looks to the user. This information will be discussed in more detail below. After the student enters his or her service number at the beginning of the program, a file is created under that number (See Appendix K, Part 5) and is used to track the student's performance. The service number, is also stored in the file STUDENT to assist in the analysis of the student data.

Figure 9 is a somewhat detailed description of the type of information stored in each record of the file POINT.ONE. The data in record #0 are used for various housekeeping functions. When the Videodisc program is started, the data contained in record #1 of POINT.ONE are loaded into the program (See Appendix K, Part 6). These data determine how the lesson will look to the user and which record of POINT.ONE will be accessed next, contingent on the choice made by the student. If the student chooses the pedagogical mode of usage,

Figure 8. Overview of Data Flow

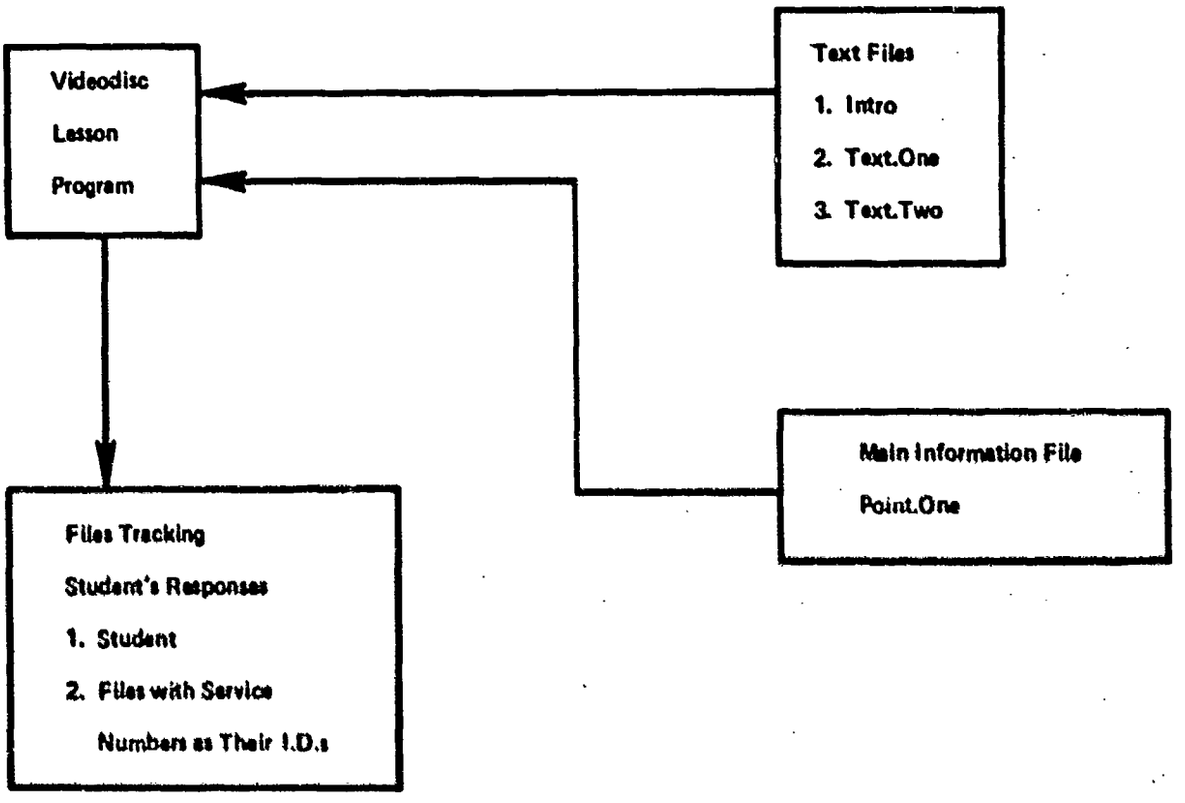


Figure 8. Flow of data into and out of Videodisc program.

Figure 9. Contents of Point.One

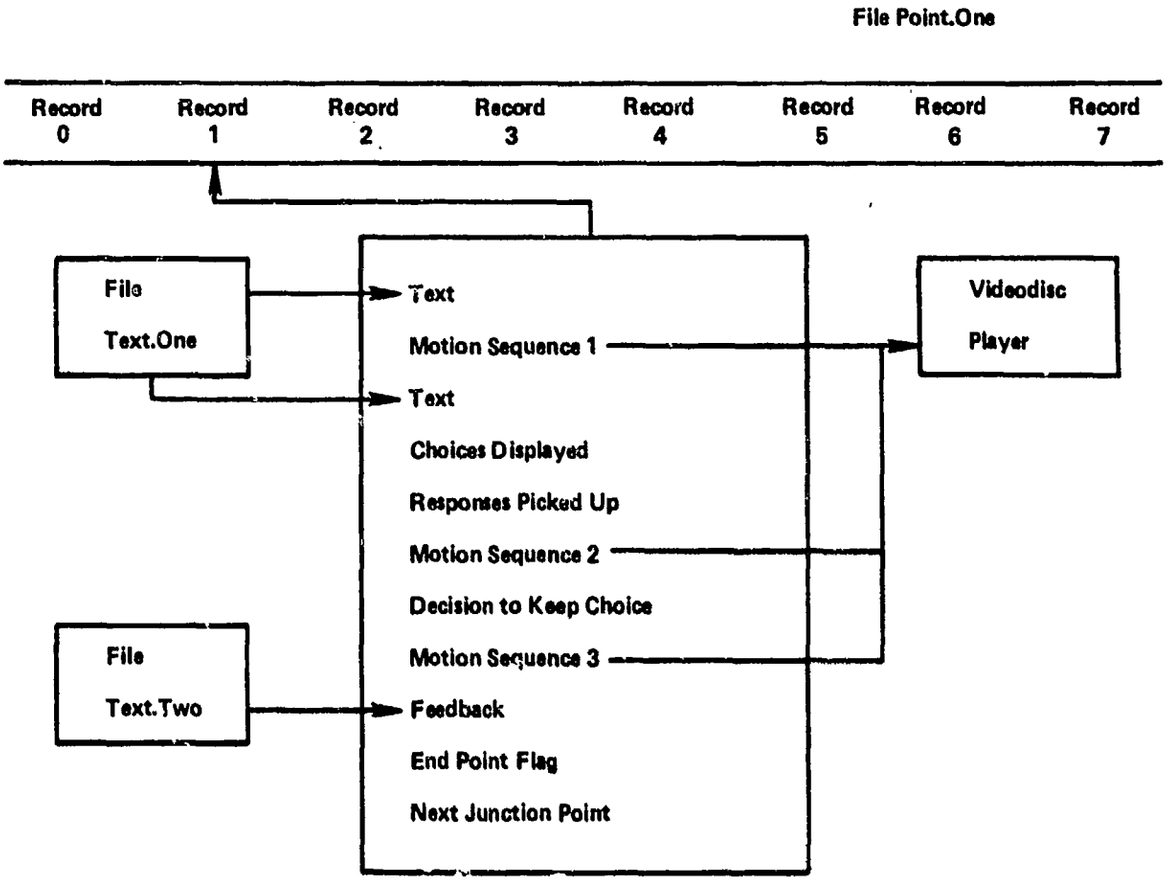


Figure 9. The types of information stored in each record of the file Point.One. Each record represents a different choice point.

the program progresses sequentially through the records of POINT.ONE.* However, if the student chooses the experiential mode, the program will usually access the records of POINT.ONE in a nonsequential manner.

Lesson Administrative Maintenance Program (LAMP)

The following is a general discussion of the system support program, LAMP. LAMP consists of two subprograms, Structure and Transfer (See Appendix K, Part 4). Each of the subprograms relies on counter files that are associated with each of the text files. The convention used to name the counter files was to add .COUNT to the name of the file. For example, the counter file for TEXT.ONE would be TEXT.ONE.COUNT.

Structure is used to fill the file POINT.ONE with the appropriate information. It is a general utility and maintenance program that permits the user to create, view, print out, and modify the data in POINT.ONE. Structure also uses counter information stored in the file POINT.ONE.COUNT.

Transfer is used in conjunction with the Pascal Editor to create and modify the text files INTRO, TEXT.ONE, TEXT.TWO, along with the text portion of POINT.ONE. The program permits the user to use the Editor to create and modify the text and then provides for the formatting of the data. The program also allows the user to transfer the data back into a raw text form for additional editing. Finally, the program allows the user to review the text in its formatted form and to manipulate the data in the counter files.

Information contained within POINT.ONE

The following is a description of the variables contained in each record of POINT.ONE. A record exists for each junction point of the lesson starting with record #1 of POINT.ONE. Record #Ø of POINT.ONE is used for housekeeping functions. A discussion of these data follows in the subsection labeled Information contained in record #Ø of POINT.ONE. Information pertaining to feedback is usually dummied for the experiential mode junction points.

SEQ1: PACKED ARRAY [1..2] OF STRING [5]: This variable contains the start and stop values of the first motion sequence of each junction point.

SEQ2: PACKED ARRAY [1..5, 1..2] OF STRING [5]: This variable contains the start and stop values of the motion sequence looking at the lieutenant for five potential choices.

SEQ3: PACKED ARRAY [1..5, 1..4] OF STRING [5]: This variable contains the start and stop values of the motion sequence looking at the person

*This is the case for the POINT.ONE files created for this contract. The data do not have to be loaded in this manner.

being counseled for five potential choices. Information for two different sequences that will be patched together by the program may be loaded into this variable. For example SEQ3[1,1] and SEQ3[1,2] contain the start and stop frame numbers for one motion sequence for answer #1. While SEQ3[1,3] and SEQ3[1,4] contain the same information for the second motion sequence.

ANSWERS: PACKED ARRAY [1..20] OF STRING [40]: This variable contains the text of the choices available to the student.

POSITIONS: ARRAY [1..5, 1..2] OF INTEGER: This variable contains the beginning and end line numbers of the text for five potential answers.

SEQMANY: PACKED ARRAY [1..5] OF INTEGER: This variable contains the number of motion sequences to be used to interpret SEQ3. This number must either be 1 or 2.

FEEDMANY: PACKED ARRAY [1..5] OF INTEGER: This variable contains the number of pages of feedback for five potential choices.

FEEDWHERE: PACKED ARRAY [1..5] OF INTEGER: This variable contains the location (the record number) of the feedback for five potential choices. This text is contained in file TEXT.TWO.

CORRECT: INTEGER: This variable contains the correct choice number for this junction point.

TEXTMANY: PACKED ARRAY [1..2] OF INTEGER: This variable contains the number of pages of text to be displayed before and after the first motion sequence. TEXTMANY [1] refers to text before and TEXTMANY [2] refers to text after the first motion sequence.

TEXTWHERE: INTEGER: This variable contains the location (the record number) of text to be displayed before and/or after the first motion sequence of a junction point. This text is contained in file TEXT.ONE.

RESPONSE: BOOLEAN: This variable indicates whether or not choices should be displayed and a response picked up. This variable is usually set to TRUE.

NEXT: PACKED ARRAY [1..5] OF INTEGER: This variable contains the location of the next record to be accessed in file POINT.ONE given that a particular response has been made.

DONE: PACKED ARRAY [1..5] OF BOOLEAN: This variable indicates whether or not a particular choice results in the termination of the program.

TEXT: PACKED ARRAY [1..2, 1..2] OF BOOLEAN: This variable indicates whether or not text is displayed before and/or after the first motion sequence of a junction point. This variable is dimensioned such that separate variables exist for the two modes of lesson usage. That is, TEXT [1..2, 1] refers to the pedagogical mode, and TEXT [1..2, 2] refers to the experiential.

PLAY: PACKED ARRAY [1..2, 1..2] OF BOOLEAN: This variable indicates whether or not the first and/or third motion sequences will be displayed. This variable is dimensioned so that separate variables exist for the two modes of lesson usage. (Refer to Table I for a review of the functions of the variables.)

Information stored in record #Ø of file POINT.ONE

Record zero of file POINT.ONE is used for various housekeeping functions. The variables and their functions are listed below:

SEQ1: PACKED ARRAY [1..2] OF STRING [5]: This variable is used to store the start and stop frame numbers of the light pen instructions.

TEXTMANY: PACKED ARRAY [1..2] OF INTEGER:

TEXTMANY [1]: This variable contains the number of records that constitute the file INTRO.

TEXTMANY [2]: This variable contains the number of student files that have been used so far. The identification numbers (the service numbers) of the files are stored in the file STUDENT. This variable must be set to zero for a new set of files.

TEXTWHERE: This variable contains the record in which text starts for the introductory text. This text is in file INTRO.

RESPONSE: BOOLEAN: If this variable is set to TRUE, the student will be allowed to choose which of the two modes of instruction he wishes to use. If this variable is set to FALSE, the student is forced to use the pedagogical mode of instruction.

Appendix K
Part 2

Instructional User's Guide

Before you attempt to load data for use by the Videodisc program, you should first read the following in the order that they are listed.

1. ALL Apple Pascal manuals dealing with the Filer, Editor, and the use of the Formatter program

2. Software Description

3. Functional Software Overview

4. Instructions for Lesson Administrative Maintenance Program (LAMP)

5. Instructions for Transfer

6. Instructions for Structure

7. Data Entry Overview

Before you attempt to use the program Videodisc, you should first read the following in the order that they are listed.

1. ALL Apple Pascal manuals dealing with the use of Pascal on a system containing two floppy disk drives, how to execute a program, and use of the Pascal Filer

2. Software Description

3. Functional Software Overview

4. Additional Support Software

5. Student File Format

6. Instructions for running Videodisc

Appendix K
Part 3

Additional Support Software

Four additional programs, Align, Clock, Look, and Reset are supplied on each system diskette. Each will be discuesed in detail below.

ALIGN

Align is used to align the light pen for use with Videodisc. Remove the cover of the Apple and identify the light pen card before you execute the program. Note that there are three screw adjustments on the top of the card. The adjustment closest to the keyboard is the Y axis adjustment and the next screw back is the X axis adjustment. The adjustment screw in the very back is a sync adjustment and SHOULD NOT be altered unless ABSOLUTELY NECESSARY! Execute the program Align and touch the pen to a location on the screen. The X and Y positions will be displayed at the top left corner of the screen. If these values do not match the values that you have selected with your positioning, CAREFULLY AND SLOWLY make the proper adjustment. Select line 20 to exit the program after the pen is correctly adjusted.

CLOCK

Clock is used to reset the clock when necessary. Execute the program and select S to set the clock. Press return to the questions regarding the WRITE ENABLE and LEAP YEAR switches. Answer with a Y to the LEAP YEAR question. Enter the correct information regarding month, date, hour (in Creenwich Time), minutes, and seconds. Answer with a return to the statement about setting the enable switch. Press D to display the time. If it is not correct, repeat the above instructions. If it is correct, press Q to exit the program.

LOOK

Look is used to review or print out the data contained in all of the student records that have been created (See Appendix K, Part 5). Before you execute the program, insure that the diskette that contains the student files which you wish to review is in device #5 (floppy disc drive #2). After you execute the program, you will be asked if you wish to use the printer. Answer with a Y or N to this question. If you answer with a Y, the program will print out the data contained in all of the student files contained on the diskette in device #5. If you answer with a N, the program will display the preliminary data for the first file. Press return to continue the display. You must press the return to review all subsequent segments of information.

RESET

Reset is used to reset variable TEXTMANY [2] in record #Ø of file POINT.ONE to zero. This program will also remove all student files from the diskette in device #5 after TEXTMANY [2] is reset. Before you execute the program, insure that you have printed out the information contained in all of the student's file if you wish to preserve this information. Next, place the diskette containing the POINT.ONE file to be reset in device #5. Finally, execute the program. If POINT.ONE has been reset, the program will inform you of this. Otherwise, the program will display that it has reset POINT.ONE and that it is removing the student files on the diskette.

Appendix K
Part 4

Instructions for Lesson Administrative Maintenance Program

The Lesson Administrative Maintenance Program, or LAMP, allows the user to create, fill, and modify all the data files used by the Videodisc lesson program (See Appendix K, Part 6). The data files accessed by this program are INTRO, TEXT.ONE, TEXT.TWO, POINT.ONE and the counter files associated with each of these files. The convention for naming the counter files is to add .COUNT to their associated file name such that the counter file for INTRO is INTRO.COUNT. These counter files contain a single integer that represents the actual number of records in each file. Figure 10 diagrams the organization of LAMP.

LAMP is a menu driven program. A menu driven program is one that displays a list of options whenever any major action is available to the user. A particular option may then be chosen by "entering" the number associated with that option. A number is "entered" by typing that number and then pressing the return key.

When LAMP is executed, the following menu is displayed:

1. RUN "TRANSFER"

2. RUN "STRUCTURE"

3. EXIT TO EDITOR

4. EXIT PROGRAM

Options 1 and 2, RUN "TRANSFER" and RUN "STRUCTURE", allow access to the two major portions of the LAMP program. Option 1 is used to facilitate the creation, input of data, and manipulation of data in the text files INTRO, TEXT.ONE, TEXT.TWO, and the text portion of POINT.ONE (See Table 27). Option 1 also allows the viewing and printing of these files in their formatted form, and for the creation and manipulation of the counter files. Option 2 is used to create, fill, and manipulate the POINT.ONE or main data file.* Both Options 1 and 2 are explained in more detail later.

Option 3, EXIT TO EDITOR, allows the user to exit directly into the Pascal editor. The value of this option will be more obvious later as Option 1 assumes that the Pascal editor will be used to manipulate the data in the text files.

*When any file is created, it is placed last on the disk and, therefore, can be filled with no side effects. However, if the user wishes to add to a file after another file has been created, the file that is to be extended must be placed last on the disk and the disk packed to conserve room. See the Apple Pascal manuals pertaining to the Filer for more information on this procedure. There is one exception, when the option ADD TO THE FILE under Structure is chosen, the file is placed last on the disk by the program with no user intervention.

Figure 10. LAMP

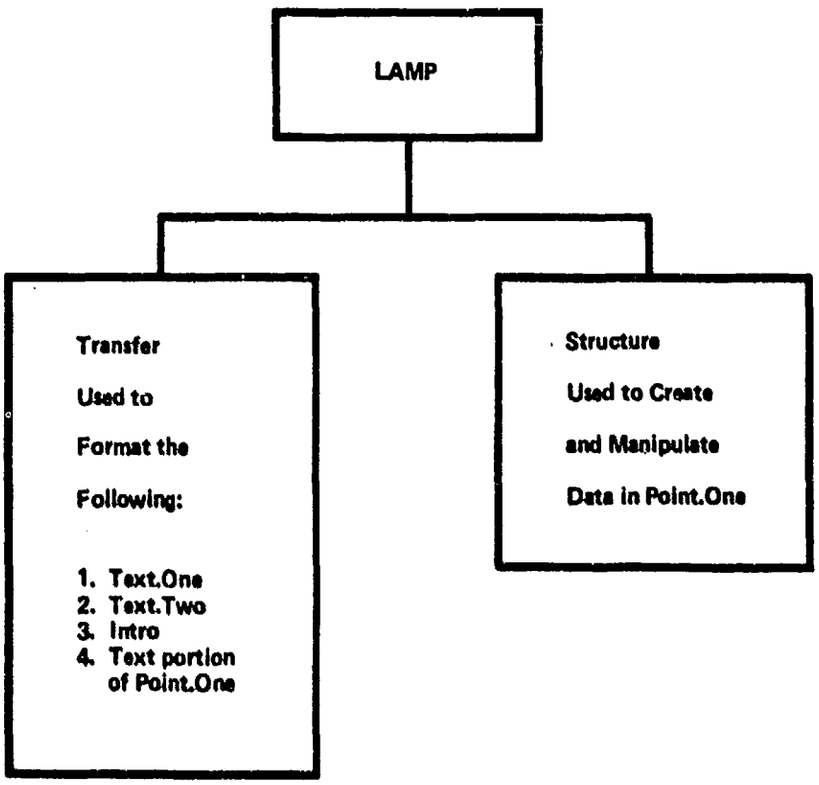


Figure 10. The organization of the LAMP program.

Table 27

Review of Variables Displayed by Options in Structure

Option	Variables that are displayed
1. Sequence One Information	SEQ1[1..2]
2. Text of Choices	ANSWERS[1..2Ø]
3. Position of Choices	POSITIONS[1..5, 1..2]
4. Correct Choice	CORRECT
5. Text Before and/or After Motion Sequence	TEXT[1..2, 1..2]
6. Number of Pages of Text	TEXT[1..2] TEXTWHERE
7. Pick up a Response	RESPONSE PLAY[1..2, 1..2]
8. Information Pertaining to Choices:	
a. Sequence Two	SEQ1[1..5, 1..2]
b. Sequence Three	SEQ3[1..5, 1..4] SEQMANY
c. Pages of Explanation	FEEDMANY[1..5] FEEDWHERE[1..5]
d. Next Junction Point	NEXT[1..5] DONE[1..5]

After completing Options 1 or 2, the user may only exit back to the LAMP main menu so that all entry and exit points are at the LAMP main menu. The only way to correctly exit the LAMP program is through Option 4, EXIT PROGRAM.

Instructions for Transfer

Transfer is used to facilitate the input of text into the files TEXT.ONE, TEXT.TWO, INTRO, and POINT.ONE. The program also allows viewing the data in these files in their formatted form. Finally, the user can create and manipulate the data in the counter files. A counter file must exist for each of the information files. The convention used for naming counter files is to add .COUNT to the file name. Therefore, the counter file for INTRO would be INTRO.COUNT. Transfer automatically creates counter files for you, as well as updating them.

INTRO contains the text of the overview that is presented to the student at the beginning of the lesson. This overview describes how the program functions in both the pedagogical and experiential modes. This information must be entered in the sequence in which it will be displayed. TEXT.ONE contains the text that is displayed before and/or after the first motion sequence of a junction point. This text is optional for any given junction point and is entered as it will be displayed at each junction point. However, the text for the various junction points can be entered in any order, i.e., the text for junction point 4 could be entered first and the text for junction point 1 could be entered next, etc.

TEXT.TWO contains the text that is displayed after sequence three as feedback in the pedagogical mode. The text for any given junction point and choice within that junction point must be entered sequentially, but the order of the entry for all choices and junction points does not depend on order. For example the text for choice point one, choice three may be entered after choice point two, choice one.

The user first develops the appropriate text for a particular file using the Pascal Editor. The Pascal Editor is utilized because of its powerful editing features. This will simplify and expedite the entry and modification of textual information. After all changes have been made, Transfer is used to format the text for use by the videodisc program. Transfer, as well as Videodisc, assumes that a page of text consists of no more than 20 lines of text with a maximum of 40 characters per line. Transfer also assumes that each page of text begins with the word START and ends with the word END. Finally, the last page of text must also have the word FINISHED following START, END, and FINISHED must be flush with the left-hand margin and have no other characters or blanks to the right. Figure 11 is an example of this format. Any information that does not conform to this format, for example, a line longer than forty characters, will be lost during the transfer process.

Figure 11. Transfer Format

Left Hand Margin

START

Maximum of 40 characters to a line

Maximum of 20 lines of text to a page

END

START

Text

END

FINISHED

Figure 11. Example of screen format for Transfer program.

When Transfer is executed, the following options are available:

1. MOVE FORMATTED FILES TO A TEXT FILE
2. MOVE A TEXT FILE TO A FORMATTED FILE
3. MOVE POINT.ONE TO A TEXT FILE
4. MOVE A TEXT FILE TO POINT.ONE
5. REVIEW FILES IN FORMATTED FORM
6. CHECK COUNTER FILES
7. EXIT PROGRAM

MOVE FORMATTED FILES TO A TEXT FILE: This option is used to move the formatted data files into an unformatted text file that can be accessed by the Pascal Editor. When this option is chosen, a new menu is displayed, which is:

1. SOURCE FILE TEXT.ONE ON DRIVE #5
2. SOURCE FILE TEXT.TWO ON DRIVE #5
3. SOURCE FILE INTRO ON DRIVE #5
4. OTHER
5. RETURN TO MAIN MENU

Options 1 through 3 result in the appropriate file on Drive #5 (Disk Drive 2) being used for transfer to an unformatted file. These options assume that the counter file for the selected file also exists on Drive #5. If you wish to access other files or if the associated counter file is not on Drive #5 then Option 4, OTHER, must be chosen. This will allow you to enter the desired Drive # and file name. If you wish to return to the main menu, choose Option 5. The only way to exit this program is through the main menu.

The program will then ask you to enter the drive number and name of the destination file. This must be of the form - DRIVE NUMBER: NAME. When the program asks you to ENTER a message, the return key must be pressed for the program to read the message.

After this is done, the program will display how many records are in the file and will ask if you want to transfer all records. If you answer with a Y, the program will proceed to transfer all records. If you answer with N, the program will ask you for the start and stop record numbers. The Pascal Editor has limitations on the size of a file it can work with, therefore, transfer only the data that are necessary. The program then indicates that the data are being transferred and which records have been transferred. After the records have been transferred, the program returns to the main menu.

MOVE A TEXT FILE TO A FORMATTED FILE: This option allows a text file to be transferred to a file formatted for use by Videodisc. When this option is chosen, a menu is displayed, which is:

1. DESTINATION FILE TEXT.ONE ON DRIVE #5
2. DESTINATION FILE TEXT.TWO ON DRIVE #5
3. DESTINATION FILE INTRO ON DRIVE #5
4. OTHER
5. RETURN TO MAIN MENU

For Options 1 through 3, the information in the source file will be placed into the respective data file. If the destination file has a different name or it is on another Drive, the "OTHER" Option must be used and the file name entered. You will then be asked to enter the drive number and the name of the source file. This file must have been created previously by the Pascal Editor. Next, you will be asked to enter the destination record. If this is a new file, enter Ø. You are then asked if the file exists. If you answer Y, the file will be opened for inputting the data. If you answer N, the text file and its associated counter file are created.

MOVE POINT.ONE TO A TEXT FILE: After choosing this option, you will be shown a two-choice menu. If you wish to return to the main menu, choose Option 2; otherwise, choose Option 1. After choosing Option 1, you will be asked if the source file is #5: POINT.ONE. If it is not, enter the source drive and file name. You will then be asked to enter the destination drive number and file name. This must be of this form - DRIVE NUMBER: NAME, where NAME is any name not already resident on that disk. The number of records is then displayed and you will be asked if you want all the records transferred. If you respond with a Y, all the records will be transferred. If you respond with N, you will be asked to enter the start and stop record numbers. The text in each designated record of POINT.ONE will then be transferred to the text file with the data delimited by the words START and END.

MOVE A TEXT FILE TO POINT.ONE: After selecting this choice, you will be shown a two-choice menu. If you choose selection 2, you will return to the main menu. If you choose selection 1, you will be able to move the text from an unformatted file to the text portion of POINT.ONE, i.e., the text, that constitutes the student's choices. You will first be asked to enter the drive number and file name of the source file. Enter this in the usual manner. You will then be asked if the destination is POINT.ONE on Drive #5. If you answer N, you must enter the destination file and drive number. If you answer Y, you will be asked to enter the destination record. The data will then be transferred beginning with the destination record that you specified and will continue until all the data in the unformatted file are transferred.

REVIEW FILES IN FORMATTED FORM: This option is used to view the text of the data files in their formatted form. When this option is chosen, a new menu is displayed, which is:

1. SOURCE FILE TEXT.ONE ON DRIVE #5

2. SOURCE FILE TEXT.TWO ON DRIVE #5

3. SOURCE FILE INTRO ON DRIVE #5

4. SOURCE FILE POINT.ONE ON DRIVE #5

5. OTHER

6. RETURN TO MAIN MENU

Options 1 through 4 result in the appropriate file on Drive #5 (Disk Drive 2) being viewed in its formatted form. These options assume that the counter file for the selected file also exists on Drive #5. If you wish to access other files, then Option 5, OTHER, must be chosen. This will allow you to enter the desired drive number and file name. If you wish to return to the main menu, choose Option 6. The only way to exit this procedure is through this option.

The program will then display a new menu, which is:

1. REVIEW AN ENTIRE FILE

2. REVIEW PART OF A FILE

3. PRINT OUT AN ENTIRE FILE

4. PRINT OUT PART OF A FILE

If Option 1 is chosen the program will display an entire page of text starting with record number Ø. At the bottom of each display, the program will ask you if there are any changes to be made. This is for minor changes that can be made in the text. If you answer with a Y, you will be asked to enter the line number you wish to change. You may then enter the information for that line. The updated version of the formatted text is then displayed. If you answer N to the question, you will then be shown the next page of text and continue until all the text has been displayed.

Option 2 is similar to Option 1, except that you may view specific pages of text. After choosing this option you will be asked to enter the starting and ending page numbers. The pages are numbered from Ø. The ending number must be greater than or equal to the starting number; if this is not the case, an error message will be displayed.

If Option 3 is chosen, the entire file will be printed in its formatted form. The record number will be printed at the top of each page. Naturally, ensure that the printer is operational.

Option 4 is similar to Option 3, except that you may print specific pages of text.

CHECK COUNTER FILES: After selecting this choice, the following menu will be displayed:

1. SOURCE TEXT.ONE.COUNT ON DRIVE #5
2. SOURCE TEXT.TWO. COUNT ON DRIVE #5
3. SOURCE INTRO.COUNT ON DRIVE #5
4. SOURCE POINT.ONE.COUNT ON DRIVE #5
5. OTHER
6. RETURN TO MAIN MENU

Options 1 through 4 will allow access to the appropriate files. Option 5 will allow access to other counter files (See Appendix K, Part 1), and Option 6 will return you to the main menu.

After choosing any option but 6, you will be asked if you want to create a new file; answer N if the file already exists. If you answer Y, you will be asked for the new value. Otherwise, you will be shown the current value and asked if you wish to change that value. If you answer Y, you may enter the new value and return to the main menu. If you answer N, you will return to the main menu.

EXIT PROGRAM

Choosing this option is the only way to correctly exit this portion of the program. After choosing this option, the user will be returned to the LAMP main menu.

Instructions for Structure

Structure is used to fill the main information file, POINT. ONE, with data. When this sub-program is executed from the Lesson Administrative Maintenance Program (LAMP), the following menu of choices is displayed:

1. CREATE A NEW SET OF FILES
2. LOOK AT ALL OF THE DATA
3. LOOK AT SPECIFIC RECORDS
4. PRINT OUT ALL OF THE DATA
5. PRINT OUT SPECIFIC RECORDS
6. ADD TO THE FILE
7. EXIT THE PROGRAM

Each of these options is discussed in detail below. A carriage return is used to complete each entry of data. A carriage return can be used to answer no to a yes/no question.

CREATE A NEW SET OF FILES: This option is used to create a new POINT.ONE file and its associated counter file, POINT.ONE.COUNT. When this option is selected, a counter file is first created and then a new POINT.ONE file is opened. You will then be permitted to fill the file with data.* After the file is opened, you will be asked to enter the following information:

1. START AND STOP FRAME NUMBERS FOR SEQUENCE 1

2. CORRECT CHOICE NUMBER

3. WHETHER OR NOT THERE IS TEXT BEFORE AND/OR AFTER SEQUENCE 1

4. THE NUMBER OF PAGES OF TEXT BEFORE AND/OR AFTER SEQUENCE 1

After you have entered the data for the number of pages of text before and/or after sequence 1, the program will ask if you wish to "dummy" the data. This question refers to the specific information, frame numbers for sequence 2, sequence 3, pages of feedback, which record contains the feedback, the next junction point, and whether a particular choice results in an end point being reached for each of the five potential answers. It is sometimes more efficient to dummy this data and change it later. After you have dummied the data or have entered it, the program will ask if you are finished entering data. If you answer with a Y, you will be returned to the menu for Structure. If you answer with any other letter, you will be permitted to enter data into the next record of POINT.ONE.** You will notice that you are not asked to enter the text of the choices, their associated start and stop line numbers, and variables pertaining to whether a response should be picked up. This is done because the text is dummied for you and the start and stop line numbers are set to the number 40. The organization of Structure was built on the assumptions that the Pascal Editor would be used to create text for POINT.ONE and that Transfer would be used to move the text into the appropriate record within POINT.ONE. The user can then, easily use the review and alter capability of Structure to change the line numbers to their correct values. The variables pertaining to a response being picked up are all set to TRUE because this is their normal state. Again, the user can go back and change them if necessary.

*The first record that you will enter data into is record #0. This is the record that contains housekeeping information for VIDEODISC. Refer to the subsection labeled Information stored in record #0 of POINT.ONE in the Functional Software Overview section for information regarding what variables are used. NOTE: The variable TEXTMANY[2] must be set to zero for a new record. RESET is used to reset this field to zero after the videodisc program has been run.

**NOTE: The next record, record #1, is the record that contains information pertaining to the first junction point of the scenarios. (See Appendix K, Part 1).

LOOK AT ALL OF THE DATA: This option allows you to review data contained in all of the records of POINT.ONE.

Reviewing specific data. After the file is opened, you will be asked if you wish to execute all of the available procedures. If you answer with a Y, you will review all of the data in all of the records. Because there is a great amount of information stored in each record, you will usually answer no to this question by typing any letter other than a Y. If you choose to not review all of the data, the following menu will be displayed:

1. SEQUENCE ONE INFORMATION
2. TEXT OF CHOICES
3. POSITION OF CHOICES
4. CORRECT CHOICE
5. TEXT BEFORE AND/OR AFTER MOTION SEQUENCE
6. NUMBER OF PAGES OF TEXT
7. PICK UP A RESPONSE
8. INFORMATION PERTAINING TO ANSWERS

Each of the available options will be discussed below. Each option will allow you to review and/or alter data displayed by the option.

1. SEQUENCE ONE INFORMATION: This option displays the start and stop frame numbers for the first motion sequence of a junction point.

2. TEXT OF CHOICES: This option displays the text of the choices that will be displayed to the student.

3. POSITION OF CHOICES: This option will display the start and stop line numbers for five potential answers. The text of the answers will also be displayed at the same time to facilitate the entering of the correct information.

4. CORRECT CHOICE: This option will display the correct choice for this junction point.

5. TEXT BEFORE AND/OR AFTER MOTION SEQUENCE: This option will display the variables that determine whether or not text is displayed before and/or after the first motion sequence of a junction point.

6. NUMBER OF PAGES OF TEXT: This option will display the number of pages of text to be displayed before and/or after the first motion sequence.

7. PICK UP A RESPONSE: This option will display the variables associated with picking up a response as well as the variables associated with playing the the first and/or third motion sequences.

8. INFORMATION PERTAINING TO ANSWERS: This option will display the start and stop frame numbers for the second and third motion sequences, how many third motion sequences there are, the number of pages and the locations of feedback text, which junction point should be accessed next, and whether or not an end point has been reached for each one of the five potential answers. When this option is selected, you are asked whether or not you wish to review the data for all answers. If you answer with anything other than a Y, you will be asked to give the start and stop numbers of the answers that you wish to review. You will then be asked if you wish to review all of the data. If you answer with anything other than a Y, the following menu will be displayed:

1. SEQUENCE TWO

2. SEQUENCE THREE

3. PAGES OF EXPLANATION

4. NEXT JUNCTION POINT

Each of the above choices will be discussed below:

1. SEQUENCE TWO: This option will display the start and stop frame numbers for sequence two.

2. SEQUENCE THREE: This option will display the start and stop frame numbers for the first and second sequence three options as well as the number of third motion sequences that will be used. There will usually be only one third motion sequence unless you wish to patch together two sequences. Although this option is available to you, the poor quality that results argues against its usage.

3. PAGES OF EXPLANATION: This option will display the number of pages of feedback that exist for this answer as well as the location of this feedback. The feedback is stored in TEXT.TWO.

4. NEXT JUNCTION POINT: This option will display the next junction point to access as well as whether or not this is an end point.

The options available to the user under the choice pertaining to the information for individual answers should facilitate the initial entry of data as well as any changes that might be necessary during the development of a lesson. Table 28 is a review of the variables that are displayed under each of the options discussed above.

Table 28

Description of Variables in POINT.ONE

Variable	Function of Variable
SEQ1 [1]	Start frame number for first motion sequence
SEQ1 [2]	Stop frame number for first motion sequence
SEQ2 [1..5, 1]	Start frame number of second motion sequence for each one of five answers
SEQ2 [1..5, 2]	Stop frame number of second motion sequence for each one of five answers
SEQ3 [1..5, 1]	Start frame number of the first of two potential third motion sequences for each one of five answers
SEQ3 [1..5, 2]	Stop frame number of the first of two potential third motion sequences for each one of five answers
SEQ3 [1..5, 3]	Start frame number of the second of two potential third motion sequences for each one of five answers
SEQ3 [1..5, 4]	Stop frame number of the second of two potential third motion sequences for each one of five answers
ANSWERS [1..20]	Text of choices available to the student
POSITIONS [1..5, 1]	Beginning line number for each one of five answers
POSITIONS [1..5, 2]	Ending line number for each one of five answers
SEQMANY [1..5]	Dictates how many, one or two, third motion sequences exist for each of five answers
FEEDMANY [1..5]	Dictates the number of pages of feedback for each one of five answers
FEEDWHERE [1..5]	Contains the start record number for the feedback for each one of five answers
CORRECT	Dictates which one of five answers is correct
TEXTMANY [1]	Contains the number of pages of text to be displayed before the first motion sequence
TEXTMANY [2]	Contains the number of pages of text to be displayed after the first motion sequence

TABLE 28
(cont'd.)

Variable	Function of Variable
TEXTWHERE	Contains the start record for the text to be displayed before and/or after the first motion sequence
RESPONSE	Dictates whether or not choices should be displayed and a response picked up
NEXT [1..5]	Contains the junction point to access next for each one of five answers
DONE [1..5]	Dictates whether or not an end point has been reached for each one of five answers
TEXT [1,1]	Dictates whether text should be displayed before the first motion sequence in the pedagogical mode
TEXT [2,1]	Dictates whether text should be displayed after the first motion sequence in the pedagogical mode
TEXT [1,2]	Dictates whether text should be displayed before the first motion sequence in the experiential mode
TEXT [2,2]	Dictates whether text should be displayed after the first motion sequence in the experiential mode
PLAY [1,1]	Dictates whether first motion sequence should be played in the pedagogical mode
PLAY [2,1]	Dictates whether third motion sequence should be played in the pedagogical mode
PLAY [1,2]	Dictates whether first motion sequence should be played in the experiential mode
PLAY [2,2]	Dictates whether third motion sequence should be played in the experiential mode

LOOK AT SPECIFIC RECORDS: This option is similar to the option for looking at all of the data with the exception that you specify to the program the start and stop records to be used for the review of data. As with the option to review all of the data, you may alter the data that you review. After you enter the start and stop records to be reviewed, the program will ask if you wish to use all procedures. If you choose to not review all of the data, a menu will be presented that will permit you to choose which datum you wish to look at. Refer to the discussion for Reviewing specific data.

PRINT OUT ALL OF THE DATA: This option will print out all of the data contained in all of the records of POINT.ONE. Before you select this option, ensure that the printer is connected to the computer. A word of warning, this option will use a large amount of paper.

PRINT OUT SPECIFIC RECORDS: This option will request the start and stop records to be printed before the program begins the print out. Before you select this option, ensure that the printer is connected to the computer and operational.

ADD TO THE FILE: This option allows you to fill the next record POINT.ONE with data. After you have filled one record with data, the program will ask if you are finished. If you answer with anything other than a Y, you will continue with the process of entering data. Refer to the sectio. for "Creating a new set of files", for a review of the types of data that you will be asked to enter.

EXIT THE PROGRAM: This option will return you to LAMP so that you can either exit LAMP, go to the sub-program Transfer, exit to the Pascal Editor, or return to Structure.

Appendix K
Part 5

Student File Format

Each student file consists of the following kinds of information:

```
FILE OF RECORD
     NUMBER:  INTEGER
     CHOSEN:  PACKED ARRAY [1..1Ø] OF INTEGER;
     TIMERS:  PACKED ARRAY [1..1Ø] OF INTEGER;
     KEEP:    PACKED ARRAY [1..1Ø] OF BOOLEAN;
END;
```

Record #Ø of each student file is used to store additional information about the student's performance. The variables and the information stored in them is given below.

NUMBER: This variable contains the number of junction points that the student passed through during the course of the lesson.

CHOSEN: This variable contains no special information.

TIMERS: The first three elements of this variable contain the start hour, minute, and seconds. The next three elements contain the stop hour, minute, and seconds.

KEEP: The first element of this variable is set to match the state of the variable PEDAGO. Consequently, it will be TRUE if the pedagogical mode was used, and FALSE if the experiential mode was used.

The data tracking the student's responses are then stored starting with record #1. Two records' worth of information are written out to the file for each junction point. This is done so that the choices made prior to and following the correct choice can be kept separate. That is, the first record written for a junction point contains the choices made prior to a correct choice being made. The second record contains the choices made, if any, after the correct choice was selected. After the last choice is made in either condition, the next element of CHOSEN to be used is set to contain zero. The type of information that is stored in each variable is discussed below.

NUMBER: This variable contains the current junction point.

CHOSEN: Each element of this variable contains the choices that the student made in the order in which they were selected.

TIMERS: Each element of this variable contains the response latency for the choices stored in CHOSEN.

KEEP: Each element of this variable indicates whether the choices, which are stored in CHOSEN, were kept or not.

Two records of information for each junction point are also stored when the student is in the experiential mode.

When the student selects the "HELP" option or when an end point is reached, the following information is stored: NUMBER is set to zero and the first element of CHOSEN is set to contain a number indicating the option that the student selected to continue with the lesson. The following code is used for this purpose:

Pedagogical Mode	Experiential Mode
1. GO BACK TO BEGINNING ?	1. GO BACK TO BEGINNING ?
2. GO BACK ONE JUNCTION POINT ?	2. GO BACK ONE JUNCTION POINT ?
3. REPEAT LAST JUNCTION POINT ?	3. REPEAT LAST JUNCTION POINT ?
4. QUIT THE PROGRAM ?	4. QUIT THE PROGRAM ?
5. IGNORE THIS HELP REQUEST ?	

The format of two records being written per junction point is also used when the student selects "HELP" or when an end point is reached; the same data are written twice.

Appendix K
Part 6

Data Entry Overview

The following is an overview of the procedures to be followed in entering data that will be used by the program Videodisc. The sequence given below is the recommended sequence. The user, however, may elect to enter the data in a different order (See Appendix K, Part 1). The steps listed below assume that the user has already developed a scenario that is complete with feedback, that a videodisc has been produced for the scenario, and that the user has determined all of the start and stop frame numbers for all of the motion sequences.

Complete a Data Sheet for the Scenario

The user should first fill out a data sheet that contains the information for each junction point of the scenario. If both the Pedagogical and Experiential modes will be used, list the Pedagogical path first followed by the Experiential junction points. Table 29 is a sample of the type of data sheet that you should use.

Enter All Text for the Scenario

After you have formatted a new floppy diskette, you should enter all of the text that the scenario requires for its operation. You may elect to transfer the files INTRO and INTRO.COUNT from an already formatted diskette (See Appendix K, Part 2) if you expect to use that file as it is. Use the Pascal Editor to create the text that will appear before and/or after the first motion sequence of a junction point in a file separate from the file that will contain the feedback for the scenario. This must be done because the text before and/or after the motion sequence will eventually go into the file TEXT.ONE, while the feedback will go into the file TEXT.TWO. Naturally, ensure that you follow the convention of text entry that the program Transfer will expect (See Appendix K, Part 4). That is, that each page of text is preceded by the word START and followed by the word END. Both START and END must be aligned with the left-hand margin and have no spaces following them. Also, each line of text must contain not more than 40 characters and each page of text must contain not more than 20 lines of text. Finally, the last page of text that is entered must have the word FINISHED following the word END that defines the end of the text for that page. You should also enter the text for all of the answers for each junction point. Each set of answers must be on a separate page. That is, each set of answers must be delimited by the words START and END.

Transfer the text into TEXT.ONE and TEXT.TWO

Execute LAMP and then select Transfer to move the text from the files created by the editor into formatted files that can be used by Videodisc. Perform this operation for the files TEXT.ONE and TEXT.TWO, but not for the file POINT.ONE because the latter file hasn't been created yet. It is recommended that you keep track of how much room remains on the diskette before you

Table 29

Sample Data Sheet for Videodisc Data

Junction Point Number							
SEQ1 start frame # stop frame #	TEXTMANY pages of text before SEQ1 pages of text after SEQ1	TEXTWHERE record # in TEXT.ONE	TEXT pedagogical text before SEQ1 ? text after SEQ1 ? experiential text before SEQ1 ? text after SEQ1 ?	PLAY pedagogical play SEQ1 ? play SEQ3 ? experiential play SEQ1 ? play SEQ3 ?			
Choice number							
1	SEQ2 start frame # stop frame #	SEQ3 first start frame # stop frame # second start frame # stop frame #	SEQMANY number of of SEQ3 sequences	FEEDMANY number of pages of feedback	FEEDWHERE record # in TEXT.TWO	NEXT next junction point to be accessed	DONE has an end point been reached ?
2	"						
3	"						
4	"						
5	"						

transfer the text into a formatted file to ensure that Transfer has enough room to fill a formatted file. If TEXT.TWO will be fairly large, move the text into TEXT.ONE first. Then remove the raw text file and perform a crunch on the diskette before you transfer the feedback text into TEXT.TWO.

Enter the data into POINT.ONE

Execute LAMP and then select Structure to create and fill the file POINT.ONE. After all of the records of POINT.ONE have been filled, return to the LAMP menu and select Transfer. Use Transfer to move the text of the answers into the proper records in POINT.ONE. Remember to start the transfer with record #1 because record #Ø is used for housekeeping functions (See Appendix K, Part 1). After all of the text has been transferred, return to the LAMP menu and select Structure again. Select Option #2 to review all of the records and then select Option #3 to review the positions of the answers. Again, all of the start and stop line numbers have been set to 40 for you. Consequently, you need only change the positions for the answers that exist for each junction point.

Execute the Videodisc Program and De-bug

Execute the Videodisc program (See Appendix K, Part 7) and return to LAMP if any changes need to be made to any of your data files. DO NOT ASSUME THAT YOUR DATA ARE CORRECT! You should spend the time to work through each possible path of the scenario to determine that everything is working correctly.

Appendix K
Part 7

Instructions for Running Videodisc

The following steps MUST be followed in the order that they are listed to prevent damage to the Apple computer and the data files.

1. Ensure that the videodisc player, computer, and monitor are properly connected. Refer to Figure I.

2. Turn on the videodisc player and load the proper videodisc, shiny side up. Close the cover of the videodisc player.

3. Insure that a SYSTEM DISKETTE is in device #4 (floppy disk drive #1).

4. Load the proper floppy diskette into device #5 (floppy disk drive #2).

5. Turn on the monitor and then the computer.

6. After Pascal has been loaded into the computer, use the Filer to enter the correct date. Exit the Filer after this has been completed.

7. Run Videodisc to ensure that the light pen ard clock are working properly. After these have been checked, press the RESET button to exit Videodisc.

8. To run Videodisc for a student, press the R button. NOTE: The file containing the student's performance data is not closed out until the student exits the program using the exit option in HELP or a normal exit at the completion of the program. If the RESET button is pressed during a lesson, the performance will be lost.

Appendix K

Part 8

Software Listings

Videodisc

```
(*$S+*)

PROGRAM FINAL;

USES LIGHTPEN,WORK,CLOCKSTUFF,VIDEODISC;

(* F IS THE MAIN POINTER FILE *)
VAR F:FILE OF RECORD
        SEQ1:PACKED ARRAY[1..2]OF STRING[5];
        SEQ2:PACKED ARRAY[1..5, 1..2]OF STRING[5];
        SEQ3:PACKED ARRAY[1..5, 1..4]OF STRING[5];
        ANSWERS:PACKED ARRAY[1..20]OF STRING[40];
        POSITIONS:ARRAY[1..5,1..2]OF INTEGER;
        SEQMANY:PACKED ARRAY[1..5]OF INTEGER;
        FEEDMANY:PACKED ARRAY[1..5]OF INTEGER;
        FEEDWHERE:PACKED ARRAY[1..5]OF INTEGER;
        CORRECT:INTEGER;
        TEXTMANY:PACKED ARRAY[1..2]OF INTEGER;
        TEXTWHERE:INTEGER;
        RESPONSE:BOOLEAN;
        NEXT:PACKED ARRAY[1..5]OF INTEGER;
        DONE:PACKED ARRAY[1..5]OF BOOLEAN;
        TEXT:PACKED ARRAY[1..2,1..2]OF BOOLEAN;
        PLAY:PACKED ARRAY[1..2,1..2]OF BOOLEAN;
      END;

    (* G IS THE STUDENT RECORD FILE *)
    G:FILE OF RECORD
        NUMBER:INTEGER;
        CHOSEN:PACKED ARRAY[1..10]OF INTEGER;
        TIMERS:PACKED ARRAY[1..10]OF INTEGER;
        KEEP.PACKED ARRAY[1..10]OF BOOLEAN;
      END;

    (* H CONTAINS FEEDBACK FOR THE PEDAGOGICAL MODE *)
    (* L CONTAINS TEXT TO BE DISPLAYED BEFORE       *)
    (* AND/OR AFTER A MOTION SEQUENCE               *)
    H:FILE OF PACKED ARRAY[1..20]OF STRING[40];
    L:FILE OF PACKED ARRAY[1..20]OF STRING[40];
    (* M CONTAINS THE NUMBERS [STORED AS STRINGS]   *)
    (* OF THE STUDENT FILES CREATED. F^.TEXTMANY[2] *)
    (* CONTAINS THE NUMBER OF FILES CREATED         *)
    M:FILE OF PACKED ARRAY[1..1]OF STRING[14];
    TIMERS:PACKED ARRAY[1..3]OF INTEGER;
    CHOICE, COUNT,  PREVIOUS, I, J, K, X, Y:INTEGER;
    PICKED, P, COUNTER, SELECTED:INTEGER;
    KEPT:PACKED ARRAY[1..5]OF INTEGER;
    JUNCTION:PACKED ARRAY[1..20]OF INTEGER;
    ANSWER:CHAR;
    PEDAGO, FOUND, HELPFLAG:BOOLEAN;
    TEMP, START, STOP:STRING;

SEGMENT PROCEDURE STARTA;
BEGIN
  (* CHECK TO SEE THAT THERE IS ROOM FOR THIS STUDENT FILE *)
  (* IF NOT, EXIT THE PROGRAM                              *)
  ERASE;
  RESET(F,'#5:POINT.ONE');
```

```
  SEEK(F,0);
  GET(F);


  IF F^.TEXTMANY[2]=62 THEN
    BEGIN
      CLOSE(F,LOCK);
      GOTOXY(0,12);
      WRITELN('NOT ENOUGH FOR THIS STUDENT RECORD');
      WRITELN;
      WRITELN('RUN THE PROGRAM TO RESET POINT.ONE');
      EXIT(PROGRAM);
    END;
  FOR I:=1 TO 10 DO
    BEGIN
      G^.CHOSEN[I]:=0;
    END;
  JUNCTION[1]:=1;
  GOTOXY(0,10);
  WRITELN(' ':3,'THE VIDEODISC IS BEING POSITIONED');
  GOTOXY(0,12);
  WRITELN(' ':14,'PLEASE WAIT');
  (* HIDE THE CURSOR *)
  GOTOXY(42,0);
  (* POSITION THE VIDEODISC ON FRAME 100      *)
  (* AND GET THE START AND STOP FRAME NUMBERS *)
  (* FOR THE LIGHT PEN INSTRUCTIONS           *)
  VIDINIT;
  WAIT;
  FIND('100');
  WAIT;
END;

(* THE NEXT PROCEDURE TICKS DOWN THE TIME AND DISPLAYS *)
(* THE LIGHT PEN INSTRUCTIONS AFTER TEN SECONDS        *)
SEGMENT PROCEDURE ORIGIN;
VAR HOLD,CURRENT:INTEGER;
BEGIN
  GOTOXY(0,8);
  WRITELN(' ':1,'IF YOU DO NOT KNOW HOW TO USE A LIGHT');
  WRITELN;
  WRITELN(' ':3,'PEN, SIT AND DO NOTHING, OTHERWISE');
  WRITELN;
  WRITELN(' ':3,'POINT THE PEN TO ONE OF THESE LINES');
  WRITELN;
  WRITELN(' ':8,'IN THE NEXT    SECONDS');
  GOTOXY(42,0);
  (* USE 'TIMESTAMP' OR 'READTIME' DEPENDING  *)
  (* ON THE CLOCK CARD IN USE. THE FOLLWOING  *)
  (* IS FOR A MOUNTAIN HARDWARE CPS CARD      *)
  (* THE START TIME IS STORED IN TIMERS[1..3] *)
  TIMESTAMP(SYSTIME);
  WITH SYSTIME.TIME DO
    BEGIN
      TIMERS[1]:=HOUR;
      TIMERS[2]:=MINUTE;
      TIMERS[3]:=SECOND;
      HOLD:=HOUR;
```

```
      COUNT:=MINUTE*60+SECOND;
      COUNT:=COUNT+10;
    (* CONTINUE TO READ TIME AND DISPLAY IT USING 'PLACE' UNTIL *)
    (* TEN SECONDS HAVE GONE BY OR THE RING IS TOUCHED...       *)
    (* UNTIL RINGTOUCH=TRUE                                     *)


      REPEAT
        TIMESTAMP(SYSTIME);
        IF HOUR<>HOLD THEN
        MINUTE:=MINUTE+60;
        CHOICE:=MINUTE*60+SECOND;
        CURRENT:=(COUNT-CHOICE)+1;
        PLACE(CURRENT,1852);
      UNTIL (RINGTOUCH) OR (CHOICE>=COUNT);
    END;
  IF CHOICE<COUNT THEN
    BEGIN
      HUNT(X,Y);
      WRITE(CHR(7));
    END;
END;

(* THE NEXT PROCEDURE GIVES THE STUDENT A DRILL *)
(* IN USING THE LIGHT PEN                       *)
SEGMENT PROCEDURE LIGHT;
VAR PROMPT:ARRAY[1..3]OF STRING;

PROCEDURE PICK;
BEGIN
  FOR I:=1 TO 3 DO
    BEGIN
      WRITELN(PROMPT[I]);
    END;
  GOTOXY(0,42);
  HUNT(X,Y);
  WRITE(CHR(7));
  ERASE;
END;

BEGIN
  ERASE;
  PROMPT[1]:='   PRESS THE TIP OF THE PEN TO ONE OF ';
  PROMPT[2]:='  THESE LINES AND TOUCH THE BRASS RING ';
  PROMPT[3]:='          UNTIL YOU HEAR A BEEP';
  GOTOXY(0,11);
  PICK;
  GOTOXY(0,0);
  PICK;
  GOTOXY(0,20);
  PICK;
  GOTOXY(0,3);
  PICK;
  GOTOXY(0,17);
  PICK;
END;

(* THE NEXT PROCEDURE DISPLAYS THE MONTH, DAY AND TIME OF DAY *)
```

```
SEGMENT PROCEDURE TIME;
VAR CURRENT:INTEGER;
BEGIN
  ERASE;
  TIMESTAMP(SYSTIME);
  LINES(2);
  WRITE('TODAYS DATE IS: ');
  WRITE(SYSTIME.DATE.MONTH,'/',SYSTIME.DATE.DAY,'/');
  WRITELN(SYSTIME.DATE.YEAR);


  LINES(3);
  WRITELN('THE TIME IS: ');
  LINES(12);
  WRITELN('PRESS PEN TO THIS LINE TO CONTINUE');
  GOTOXY(42,0);
  (* USE 'PLACE' TO DISPLAY THE HOUR, MINUTES, AND SECONDS *)
  REPEAT;
    TIMESTAMP(SYSTIME);
    WITH SYSTIME.TIME DO
      BEGIN
        CURRENT:=SYSTIME.TIME.HOUR;
        PLACE(CURRENT,1805);
        (* USE 'POKE' TO INSERT A ':' *)
        POKE(186,1807);
        CURRENT:=SYSTIME.TIME.MINUTE;
        PLACE(CURRENT,1808);
        POKE(186,1810);
        CURRENT:=SYSTIME.TIME.SECOND;
        PLACE(CURRENT,1811);
      END;
  UNTIL RINGTOUCH;
  HUNT(X,Y);
  WRITE(CHR(7));
END;

(* THE NEXT PROCEDURE PICKS UP THE SERVICE NUMBER *)
(* AND USES IT TO CREATE THE STUDENT'S FILE       *)
SEGMENT PROCEDURE SERVICE;
VAR I:INTEGER;
    DONE:BOOLEAN;
    L, LETTER:STRING;
BEGIN
  TEMP:='';
  DONE:=FALSE;
  L:='1234567890';
  ERASE;
  WRITELN('ENTER YOUR');
  LINES(2);
  WRITELN('SERVICE NUMBER:');
  LINES(2);
  WRITE('    ');
  FOR I:=1 TO 10 DO
    BEGIN
      WRITE('  ',L[I]);
    END;
  LINES(3);
  WRITELN('PRESS PEN:');
```

```
  WRITELN;
  WRITELN('TO THIS LINE TO SUBTRACT A NUMBER');
  LINES(3);
  WRITELN('TO THIS LINE TO COMPLETE YOUR ENTRY');
  GOTOXY(42,0);
  REPEAT
    HUNT(X,Y);
    WRITE(CHR(7));


    IF Y=7 THEN
      BEGIN
        I:=X-4;
        I:=I DIV 3;
        LETTER:=COPY(L,I,1);
        TEMP:=CONCAT(TEMP,LETTER);
        IF LENGTH(TEMP)=12 THEN TEMP:=COPY(TEMP,1,11);
        IF LENGTH(TEMP)=3 THEN TEMP:=CONCAT(TEMP,' ');
        IF LENGTH(TEMP)=6 THEN TEMP:=CONCAT(TEMP,' ');
        GOTOXY(17,3);
        WRITE(TEMP,' ':2);
        GOTOXY(42,0);
      END;
    IF Y=12 THEN
      BEGIN
        I:=LENGTH(TEMP);
        IF I=4 THEN I:=3;
        IF I=7 THEN I:=6;
        I:=I-1;
        TEMP:=COPY(TEMP,1,I);
        GOTOXY(17,3);
        WRITE(TEMP,' ':5);
        GOTOXY(42,0);
      END;
    IF (Y=16) AND (LENGTH(TEMP)=11) THEN DONE:=TRUE;
  UNTIL DONE;
  TEMP:=CONCAT('#5:',TEMP);
END;
```

```
(*$I#5:SEG.CPS.TEXT*)
(*$I#5:HELP.TEXT*)

(* THE FOLLOWING IS A GENERAL CONTINUE ROUTINE *)
PROCEDURE CONT;
BEGIN
  GOTOXY(0,21);
  WRITELN('** PRESS PEN TO THIS LINE TO CONTINUE **');
  GOTOXY(42,0);
  REPEAT
    HUNT(X,Y);
  UNTIL Y=22;
  WRITE(CHR(7));
END;

(* DISPLA1 IS USED TO DISPLAY TEXT BEFORE AND/OR AFTER *)
(* THE FIRST MOTION SEQUENCE OF EACH JUNCTION POINT    *)
PROCEDURE DISPLA1(POINT:INTEGER);
BEGIN
  FOR I:=1 TO F^.TEXTMANY[POINT] DO
    BEGIN
      GET(H);
      ERASE;
      FOR J:=1 TO 20 DO
        BEGIN
          WRITELN(H^[J]);
        END;
      CONT;
    END;
END;



(* DISPLA2 IS USED TO DISPLAY FEEDBACK TO THE STUDENT *)
PROCEDURE DISPLA2;
BEGIN
  WITH F^ DO
    BEGIN
      SEEK(L,FEEDWHERE[CHOICE]);
      FOR I:=1 TO FEEDMANY[CHOICE] DO
        BEGIN
          GET(L);
          ERASE;
          FOR J:=1 TO 20 DO
            BEGIN
              WRITELN(L^[J]);
            END;
          CONT;
        END;
    END;
END;

(* DISPLA3 IS USED TO DISPLAY THE SELECTED CHOICE *)
PROCEDURE DISPLA3;
BEGIN
  WITH F^ DO
    BEGIN
```

```
        ERASE;
        FOR J:=POSITIONS[CHOICE,1] TO POSITIONS[CHOICE,2] DO
          BEGIN
            WRITELN(ANSWERS[J]);
          END;
        LINES(3);
        WRITELN(' ':4,'DO YOU WANT TO KEEP THIS ANSWER?');
      END;
  END;

  (* THE NEXT PROCEDURE PRINTS YES AND NO ACROSS THE *)
  (* SCREEN AND SETS ANSWER TO Y OR N APPROPRIATELY  *)
  PROCEDURE GETANS;
  VAR I:INTEGER;
  BEGIN
    GOTOXY(0,16);
    FOR I:=1 TO 10 DO
      BEGIN
        WRITE('YES-');
      END;
    GOTOXY(0,19);
    FOR I:=1 TO 13 DO
      BEGIN
        WRITE('NO-');
      END;
    GOTOXY(42,0);
    REPEAT
      HUNT(X,Y);
    UNTIL (Y=17) OR (Y=20);
    WRITE(CHR(7));
    IF Y=17 THEN ANSWER:='Y'
    ELSE ANSWER:='N';
  END;

  (* THE FOLLOWING CONTROLS WHETHER OR NOT THE *)
  (* OVERVIEW IS DISPLAYED                     *)
  PROCEDURE INTRO;
  BEGIN
    RESET(H,'#5:INTRO');
    RESET(L,'#5:TEXT.TWO');


    REPEAT
      ERASE;
      GOTOXY(0,4);
      WRITELN(' ':5,'WOULD YOU LIKE AN OVERVIEW OF');
      WRITELN;
      WRITELN(' ':14,'THE PROGRAM?');
      GETANS;
      IF ANSWER='Y' THEN
        BEGIN
          SEEK(H,0);
          DISPLA1(1);
        END;
    UNTIL ANSWER<>'Y';
    (* FILE H IS CHANGED FROM INTRO TO TEXT.ONE *)
    (* AND THE STUDENT FILE IS OPENED           *)
    CLOSE(H,LOCK);
```

```
    RESET(H,'#5:TEXT.ONE');
    REWRITE(G,TEMP);
    SEEK(G,1);
    (* THE STUDENT IS ALLOWED TO CHOOSE BETWEEN THE EXPERIENTIAL  *)
    (* AND THE PEDAGOGIC MODES IF F^.RESPONSE IN RECORD 0 IS TRUE *)
    (* THE VARIABLE P DENOTES WHICH MODE IS SELECTED              *)
    IF F^.RESPONSE THEN
      BEGIN
        ERASE;
        WRITELN(' ':4,'PICK ONE OF THE FOLLOWING MODES');
        GOTOXY(0,8);
        WRITELN(' ':14,'PEDAGOGICAL');
        GOTOXY(0,16);
        WRITELN(' ':14,'EXPERIENTIAL');
        GOTOXY(42,0);
        REPEAT
          HUNT(X,Y);
        UNTIL (Y=9) OR (Y=17);
        WRITE(CHR(7));
      END
    ELSE Y:=9;
    IF Y=9 THEN
      BEGIN
        PEDAGO:=TRUE;
        P:=1;
      END
    ELSE
      BEGIN
        PEDAGO:=FALSE;
        P:=2;
      END;
  END;

  (* THE FOLLOWING DISPLAYS THE CHOICES THAT HAVEN'T BEEN *)
  (* SELECTED AND PICKS UP A RESPONSE. 'CHOICE' CONTAINS  *)
  (* THE SELECTED CHOICE NUMBER. THE RESPONSE LATENCY     *)
  (* IS ALSO COMPUTED AND STORED AS G^.TIMERS[SELECTED]   *)
  PROCEDURE LOCATE;
  VAR SKIP, FINISHED:BOOLEAN;
  HOUR, TIMER, TEMP:INTEGER;
  BEGIN
    TIMESTAMP(SYSTIME);
    HOUR:=SYSTIME.TIME.HOUR;
    TIMER:=SYSTIME.TIME.MINUTE*60+SYSTIME.TIME.SECOND;
    FOUND:=FALSE;


    WITH F^ DO
      BEGIN
        (* FIRST WRITE THE PROMPT LINE IF IT EXISTS *)
        ERASE;
        TEMP:=POSITIONS[1,1]-1;
        IF TEMP <>0 THEN
          BEGIN
            FOR I:=1 TO TEMP DO
              BEGIN
                WRITELN(ANSWERS[I]);
              END;
```

```
        END;
      (* THEN, DISPLAY EACH UNSELECTED CHOICE *)
      FOR I:=1 TO 5 DO
        BEGIN
          SKIP:=FALSE;
          IF PREVIOUS >1 THEN
            BEGIN
              J:=0;
              REPEAT
                J:=J+1;
                IF KEPT[J]=I THEN SKIP:=TRUE;
              UNTIL (SKIP) OR (J=PREVIOUS);
            END;
          IF (NOT SKIP) AND (POSITIONS[I,1]<30) THEN
            BEGIN
              FOUND:=TRUE;
              TEMP:=POSITIONS[I,1];
              J:=TEMP-1;
              GOTOXY(0,J);
              FOR J:=TEMP TO POSITIONS[I,2] DO
                BEGIN
                  WRITELN(ANSWERS[J]);
                END;
            END;
          END;
        GOTOXY(42,0);
        FINISHED:=FALSE;
        IF FOUND THEN
          BEGIN
            REPEAT
              I:=0;
              HUNT(X,Y);
              REPEAT
                I:=I+1;
                IF (Y>=POSITIONS[I,1]) AND (Y<=POSITIONS[I,2])
                THEN FINISHED:=TRUE;
              UNTIL (FINISHED) OR (I=5);
            UNTIL FINISHED;
            (* STOP THE CLOCK *)
            TIMESTAMP(SYSTIME);
            IF SYSTIME.TIME.HOUR>HOUR THEN
              SYSTIME.TIME.MINUTE:=SYSTIME.TIME.MINUTE+60;
            (* SAVE THIS RESPONSE LATENCY *)
            G^.TIMERS[SELECTED]:=(SYSTIME.TIME.MINUTE*60+
              SYSTIME.TIME.SECOND)-TIMER;
            (* SET 'CHOICE' EQUAL TO THE SELECTION *)
            WRITE(CHR(7));
            CHOICE:=I;
          END;
      END;
    END;



(* THE FOLLOWING IS USED TO DISPLAY THE VARIOUS MOTION *)
(* SEQUENCES. 'WHICH' DETERMINES THE TYPE OF SEQUENCE  *)
(* THAT IS PLAYED.                                     *)
PROCEDURE PLAYIT(WHICH:INTEGER);
```

```
VAR I, J:INTEGER;
BEGIN
  WITH F^ DO
    BEGIN
      CASE WHICH OF

      1:BEGIN
          START:=SEQ1[1];
          STOP:=SEQ1[2];
          SEG(START,STOP,3);
        END;

      2:BEGIN
          START:=SEQ2[CHOICE, 1];
          STOP:=SEQ2[CHOICE, 2];
          SEG(START,STOP,3);
        END;

      3:BEGIN
          FOR I:=1 TO SEQMANY[CHOICE] DO
            BEGIN
              J:=(I-1)*2;
              START:=SEQ3[CHOICE, J+1];
              STOP:=SEQ3[CHOICE, J+2];
              SEG(START,STOP,3);
            END;
        END;
      END;
    END;
END;

(* THE FOLLOWING PICKS UP A RESPONSE AND IF THE STUDENT  *)
(* IS IN THE PEDAGOGICAL MODE, GIVES THE STUDENT THE     *)
(* OPTION OF KEEPING THE CHOICE OR PICKING ANOTHER       *)
PROCEDURE RESPOND;
BEGIN
  WITH F^ DO
    BEGIN
      REPEAT
        ANSWER:='N';
        (* GO PICK UP A RESPONSE *)
        LOCATE;
        (* IF YOU'VE NOT RUN OUT OF ANSWERS TO PICK THEN *)
        IF FOUND THEN
          BEGIN
            (* SAVE THE SELECTION *)
            G^.CHOSEN[SELECTED]:=CHOICE;
            (* IF IN THE PEDAGOGICAL MODE, ALLOW THE STUDENT *)
            (* A CHOICE AS TO WHETHER OR NOT TO KEEP THE     *)
            (* THE ANSWER THAT WAS SELECTED.                 *)
            IF PEDAGO THEN
              BEGIN
                PLAYIT(2);
                DISPLA3;
                GETANS;
                IF ANSWER='Y' THEN G^.KEEP[SELECTED]:=TRUE
                ELSE G^.KEEP[SELECTED]:=FALSE;
              END
```

```
            ELSE ANSWER:='Y';

              SELECTED:=SELECTED+1;
            END
          ELSE ANSWER:='Y';
        UNTIL ANSWER='Y';
      END;
END;

(* THE NEXT PROCEDURE GOES TO PROCEDURE 'RESPOND' AND *)
(* DISPLAYS SEQ3 IF APPROPRIATE. IT ALSO SAVES THE    *)
(* RESPONSE TO BE PLACED INTO THE STUDENT FILE        *)
PROCEDURE COUNTIT;
BEGIN
  WITH F^ DO
    BEGIN
      FOUND:=FALSE;
      IF RESPONSE THEN RESPOND
      ELSE CHOICE:=CORRECT;
      (* IF A SELECTION WAS MADE, PLAY SEQ3 IF APPROPRIATE   *)
      (* AND KEEP AN INTERNAL RECORD OF WHAT CHOICE WAS MADE *)
      IF FOUND THEN
        BEGIN
          IF PLAY[P,2] THEN PLAYIT(3);
          IF PEDAGO THEN DISPLA2
          ELSE CORRECT:=CHOICE;
          KEPT[PREVIOUS]:=CHOICE;
          PREVIOUS:=PREVIOUS+1;
        END;
    END;
END;

PROCEDURE REDO;
BEGIN
  (* 'PREVIOUS' IS USED FOR INTERNAL RECORD KEEPING *)
  (* 'SELECTED' IS USED FOR EXTERNAL RECORD KEEPING *)
  PREVIOUS:=1;
  SELECTED:=1;
  FOR J:=1 TO 5 DO
    BEGIN
      KEPT[J]:=0;
    END;
END;

(* THE FOLLOWING WRAPS THINGS UP WHEN THE STUDENT IS FINISHED    *)
(* WITH THE LESSON. FILES G, H, AND L ARE CLOSED. THE NAME OF    *)
(* THE STUDENT FILE NAME [STORED IN 'TEMP'] IS SAVED IN 'STUDENT *)
(* THE NUMBER OF SUCH FILES IS COMPUTEDAND STORED IN             *)
(* F^.TEXTMANY[2] IN RECORD 0 OF 'POINT.ONE'                     *)
PROCEDURE FINI;
BEGIN
  ERASE;
  GOTOXY(0,4);
  WRITELN(' ':13,'END OF PROGRAM');
  GOTOXY(0,21);
  WRITELN(' ':5,'PRESS PEN TO THIS LINE TO EXIT');
  GOTOXY(42,0);
```

```
  REPEAT
    HUNT(X,Y);
  UNTIL Y=22;
  WRITE(CHR(7));
  ERASE;
  (* SAVE THE TIME OF DAY IN WHICH THE STUDENT FINISHED *)
  (* THE LESSON AND SAVE THIS TIME WITH THE START TIME  *)
  TIMESTAMP(SYSTIME);


  WITH SYSTIME.TIME DO
    BEGIN
      (* SAVE THE START TIME *)
      FOR I:=1 TO 3 DO
        BEGIN
          G^.TIMERS[I]:=TIMERS[I];
        END;
      (* SAVE THE STOP TIME  *)
      G^.TIMERS[4]:=HOUR;
      G^.TIMERS[5]:=MINUTE;
      G^.TIMERS[6]:=SECOND;
    END;
  (* SAVE THE NUMBER OF JUNCTION POINTS GONE THROUGH      *)
  (* AND STORE IT IN RECORD 0 OF FILE G[THE STUDENT FILE] *)
  G^.NUMBER:=COUNTER;
  G^.KEEP[1]:=PEDAGO;
  SEEK(G,0);
  PUT(G);
  CLOSE(G,LOCK);
  CLOSE(H,LOCK);
  CLOSE(L,LOCK);
  (* FIND OUT WHERE TO PUT THE NUMBER OF THIS STUDENT *)
  (* FILE. AFTER YOU'VE STORED IT, KICK THE COUNTER   *)
  (* (F^.TEXTMANY[2]) UP BY ONE AND SAVE IT.          *)
  SEEK(F,0);
  GET(F);
  RESET(M,'#5:STUDENT');
  SEEK(M,F^.TEXTMANY[2]);
  M^[1]:=TEMP;
  PUT(M);
  F^.TEXTMANY[2]:=F^.TEXTMANY[2]+1;
  SEEK(F,0);
  PUT(F);
  CLOSE(F,LOCK);
  CLOSE(M,LOCK);
  REJECT;
END;

(* THE MAIN PROGRAM STARTS HERE *)
BEGIN
  (* INITIALIZE THE VIDEODISC AND SET VALUES *)
  STARTA;
  (* ALLOW THE STUDENT TO BYPASS THE LIGHT-PEN INSTRUCTIONS *)
  ORIGIN;
  IF CHOICE>=COUNT THEN
    BEGIN
      SEG(F^.SEQ1[1],F^.SEQ1[2],3);
      ORIGIN;
```

```
  END;
IF CHOICE>=COUNT THEN
  BEGIN
    SEG(F^.SEQ1[1],F^.SEQ1[2],3);
    LIGHT;
  END
ELSE ERASE;
(* DISPLAY THE TIME AND DATE *)
TIME;
(* PICK UP THE SERVICE NUMBER TO BE USED TO CREATE FILE G *)
SERVICE;
(* ALLOW THE STUDENT TO BYPASS THE INTRODUCTION *)
INTRO;
COUNT:=1;
COUNTER:=0;
G^.KEEP[1]:=TRUE;


REPEAT
  IF COUNT<1 THEN COUNT:=1;
  WITH F^ DO
    BEGIN
      (* SAVE THE JUNCTION POINT NUMBER *)
      G^.NUMBER:=JUNCTION[COUNT];
      (* GET THE APPROPRIATE DATA FROM FILE POINT.ONE *)
      SEEK(F,JUNCTION[COUNT]);
      GET(F);
      (* MOVE TO THE APPROPRIATE RECORD FOR TEXT *)
      SEEK(H,TEXTWHERE);
      (* DISPLAY TEXT AND MOTION SEQ1 IF APPROPRIATE *)
      IF TEXT[P,1] THEN DISPLA1(1);
      IF PLAY[P,1] THEN PLAYIT(1);
      IF TEXT[P,2] THEN DISPLA1(2);
      (* RESET COUNTERS *)
      REDO;
      (* KEEP PICKING UP CHOICES UNTIL CHOICE=CORRECT *)
      REPEAT
        COUNTIT;
      UNTIL CHOICE=CORRECT;
      (* ZERO THE NEXT G^.CHOSEN VARIABLE FOR USE BY   *)
      (* A PROGRAM FOR READING THE DATA...THIS ACTS AS *)
      (* AS AN END OF ENTRY DELIMITER                  *)
      G^.CHOSEN[SELECTED]:=0;
      (* TRANSFER THE DATA ONTO DISK *)
      PUT(G);
      REDO;
      PICKED:=CHOICE;
      (* DETERMINE WHERE YOU GO TO NEXT *)
      JUNCTION[COUNT+1]:=NEXT[PICKED];
      (* IF IN THE PEDAGOGICAL MODE, ALLOW THE FOLLOWING OPTIONS *)
      IF PEDAGO THEN
        BEGIN
          REPEAT
            HELPFLAG:=FALSE;
            ERASE;
            WRITELN(' ':8,'PRESS THE PEN TO ONE OF');
            WRITELN(' ':8,'THE FOLLOWING LINES TO:');
            GOTOXY(0,5);
```

```
                WRITELN(' ':9,'LOOK AT OTHER CHOICES?');
                GOTOXY(0,9);
                WRITELN(' ':6,'REPEAT LAST CORRECT CHOICE?');
                GOTOXY(0,13);
                WRITELN(' ':13,'REQUEST HELP?');
                GOTOXY(0,17);
                WRITELN('   CONTINUE TO THE NEXT CHOICE POINT?');
                GOTOXY(42,0);
                HUNT(X,Y);
                WRITE(CHR(7));
                IF Y=6 THEN COUNTIT;
                IF Y=10 THEN
                  BEGIN
                    IF PLAY[P,2] THEN PLAYIT(3);
                  END;
                IF Y=14 THEN
                  BEGIN
                    HELPFLAG:=TRUE;
                    Y:=18;
                  END;
              UNTIL Y=18;
            END;


          G^.CHOSEN[SELECTED]:=0;
          PUT(G);
          IF (PEDAGO) AND (HELPFLAG) THEN HELP;
          (* IF AT END POINT IN EXPERIENTIAL MODE, GO TO HELP *)
          IF (NOT PEDAGO) AND (DONE[PICKED]) THEN HELP;
          COUNT:=COUNT+1;
          COUNTER:=COUNTER+1;
        END;
    UNTIL F^.DONE[PICKED];
    (* WRAP THINGS UP *)
    FINI;
END.
```

```
(* 'SEG' DISPLAYS A PARTICULAR SEQUENCE BEGINNING WITH 'START'     *)
(* FINISHING WITH 'STOP' AND DISPLAYING THE LAST FRAME FOR 'LONG' *)
(* NUMBER OF SECONDS                                               *)

(* THIS VERSION IS FOR A MOUNTAIN HARDWARE CPS CARD                *)

PROCEDURE SEG(START, STOP:STRING;LONG:INTEGER);
VAR MATCH, HOLD, TEMP:INTEGER;
BEGIN
  FIND(START);
  WAIT;
  PLAY2(START,STOP);
  VIDEO(TRUE);
  WAIT;
  READTIME;
  TEMP:=HOURS;
  HOLD:=MINUTES*60+SECONDS;
  HOLD:=HOLD+LONG;
  REPEAT
    READTIME;
    IF HOURS>TEMP THEN MINUTES:=MINUTES+60;
    MATCH:=MINUTES*60+SECONDS;
  UNTIL MATCH=HOLD;
  ERASE;
  VIDEO(FALSE);
END;
```

```
(* 'HELP' ALLOWS THE STUDENT TO SKIP BACK A LESSON OR
   REPEAT A LESSON. *)

PROCEDURE HELP;
VAR I:INTEGER;
    TEMP:BOOLEAN;

BEGIN
  ERASE;
  IF NOT PEDAGO THEN WRITELN(' ':15,'END POINT');
  GOTOXY(0,3);
  WRITELN(' ':9,'GO BACK TO BEGINNING?');
  GOTOXY(0,7);
  WRITELN(' ':6,'GO BACK ONE JUNCTION POINT?');
  GOTOXY(0,11);
  WRITELN(' ':6,'REPEAT LAST JUNCTION POINT?');
  GOTOXY(0,15);
  WRITELN(' ':11,'QUIT THE PROGRAM?');
  GOTOXY(0,19);
  IF PEDAGO THEN WRITE(' ':7,'IGNORE THIS HELP REQUEST?      ');
  GOTOXY(42,0);
  REPEAT
    HUNT(X,Y);
    WRITE(CHR(7));
  UNTIL Y>3;
  TEMP:=F^.DONE[PICKED];
  F^.DONE[PICKED]:=FALSE;
  CASE Y OF
    4:BEGIN
        COUNT:=0;
        I:=1;
      END;

    8:BEGIN
        COUNT:=COUNT-2;
        I:=2;
      END;

    12:BEGIN
         COUNT:=COUNT-1;
         I:=3;
       END;

    16:BEGIN
         F^.DONE[PICKED]:=TRUE;
         I:=4;
       END;

    20:BEGIN
         F^.DONE[PICKED]:=TEMP;
         I:=5;
       END;

    END;
  G^.CHOSEN[1]:=I;
  G^.CHOSEN[2]:=0;
  G^.NUMBER:=0;
```

```
    PUT(G);
    G^.CHOSEN[1]:=0;
    PUT(G);
    COUNTER:=COUNTER+1;
END;
```

LAMP

```
(*$S+*)

PROGRAM LAMP;

(* THIS PROGRAM MOVES TEXT INTO AND OUT OF FORMATTED FILES       *)
(* FROM A FORM THAT CAN BE EDITED BY THE PASCAL TEXT EDITOR TO   *)
(* A FORM USED BY THE VIDEO LESSON PROGRAM. THIS PROGRAM ALSO    *)
(* HAS PROVISIONS FOR VIEWING THE FILES IN A FORMATTED FORM AND  *)
(* CHECKING THE COUNTER FILES. THIS PROGRAM ALSO IS USED TO      *)
(* CREATE, CHECK, AND CHANGE THE MAIN DATA FILE. THIS PROGRAM    *)
(* IS MENU DRIVEN WITH PROVISIONS FOR THE MOST USED FILES.       *)
(* THESE FILES ARE TEXT.ONE, TEXT.TWO, INTRO, POINT.ONE AND      *)
(* COUNTER FILES FOR EACH.                                       *)

USES LIGHTPEN, WORK, CHAINSTUFF;

(*   F IS FILE POINT.ONE  *)
VAR F:FILE OF RECORD
        SEQ1:PACKED ARRAY[1..2]OF STRING[5];
        SEQ2:PACKED ARRAY[1..5, 1..2]OF STRING[5];
        SEQ3:PACKED ARRAY[1..5, 1..4]OF STRING[5];
        ANSWERS:PACKED ARRAY[1..20]OF STRING[40];
        POSITIONS:ARRAY[1..5,1..2]OF INTEGER;
        SEQMANY:PACKED ARRAY[1..5]OF INTEGER;
        FEEDMANY:PACKED ARRAY[1..5]OF INTEGER;
        FEEDWHERE:PACKED ARRAY[1..5]OF INTEGER;
        CORRECT:INTEGER;
        TEXTMANY:PACKED ARRAY[1..2]OF INTEGER;
        TEXTWHERE:INTEGER;
        RESPONSE:BOOLEAN;
        NEXT:PACKED ARRAY[1..5]OF INTEGER;
        DONE:PACKED ARRAY[1..5]OF BOOLEAN;
        TEXT:PACKED ARRAY[1..2,1..2]OF BOOLEAN;
        PLAY:PACKED ARRAY[1..2,1..2]OF BOOLEAN;
      END;

    (* G IS THE TEXT FILE USUALLY A FILE USED BY THE PASCAL EDITER *)
    G:INTERACTIVE;

    (* L IS THE FORMATTED FILE USUALLY INTRO, TEXT.ONE, OR TEXT.TWO*)
    L:FILE OF PACKED ARRAY[1..20] OF STRING[40];

    (* H IS THE COUNTER FILE USUALLY INTRO.COUNT, TEXT.ONE.COUNT,  *)
    (*          TEXT.TWO.COUNT, OR POINT.ONE.COUNT                 *)
    H:FILE OF INTEGER;

    COUNTER, ORIGIN, EXTENT, VALUE, COUNT, I, J, K:INTEGER;
    BIGGY, WHICH, START, STOP:INTEGER;
    PROMPT:PACKED ARRAY[1..2]OF STRING[40];
    ANS, ANSWER:CHAR;
    COUNTSTR, SOURCE, DESTINATION, DUMMY,RESPONSE:STRING;
    LEAVE, ADD, EVERY, SPECIFIC, LIST, FEEDBACK, CHANGE, LOOK, NEW:BOOLEAN;
    OUT:INTERACTIVE;

(*$I#5:FOIST.TEXT*)
(*$I#5:TRANSFER.TEXT*)

(*$I-*)
```

```
(*$I#5:FIRST.TEXT*)
(*$I#5:SECOND.TEXT*)
(*$I#5:STRUCTURE.TEXT*)

BEGIN
 POKE(0,-16300);
  REPEAT
    ERASE;
    LINES(2);
    WRITELN('INFORMATION FILE MAINTAINANCE PROGRAM');
    LINES(2);
    WRITELN('ENTER YOUR CHOICE-->');
    LINES(3);
    WRITELN('  1. RUN "TRANSFER"');
    WRITELN;
    WRITELN('  2. RUN "STRUCTURE"');
    WRITELN;
    WRITELN('  3. EXIT TO EDITOR');
    WRITELN;
    WRITELN('  4. EXIT PROGRAM');
    GOTOXY(20,5);
    READLN(BIGGY);
    GOTOXY(42,0);
    CASE BIGGY OF
      1:TRANSFER;
      2:STRUCTURE;
      3:BEGIN
          RESPONSE:='';
          SETCVAL(RESPONSE);
          SETCHAIN('EXEC/EDT.TEXT');
          EXIT(PROGRAM);
        END;
      4:BEGIN
          ERASE;
          EXIT(PROGRAM);
        END;
    END;
  UNTIL BIGGY=4;
END.
```

```
SEGMENT PROCEDURE TRANSONE;

PROCEDURE BREAKUP;

BEGIN
 REPEAT (* LOOP UNTIL CORRECT NUMBER ENTERED *)
  ERASE;
  WRITELN;
  WRITELN('TRANSFER A FORMATTED FILE TO AN');
  WRITELN;
  WRITELN('UNFORMATTED FILE');
  GOTOXY(0,6);
  WRITELN('ENTER YOUR CHOICE-->');
  LINES(2);
  WRITELN('1. SOURCE FILE TEXT.ONE ON DRIVE #5');
  WRITELN;
  WRITELN('2. SOURCE FILE TEXT.TWO ON DRIVE #5');
  WRITELN;
  WRITELN('3. SOURCE FILE INTRO    ON DRIVE #5');
  WRITELN;
  WRITELN('4. OTHER');
  WRITELN;
  WRITELN('5. RETURN TO MAIN MENU');
  GOTOXY(20,6);
  READLN(WHICH);
  GOTOXY(42,2);
  ERASE;

   CASE WHICH OF

     1:SOURCE:='#5:TEXT.ONE';
     2:SOURCE:='#5:TEXT.TWO';
     3:SOURCE:='#5:INTRO';
     4:BEGIN
        WRITELN;
        WRITELN('ENTER DRIVE NUMBER AND');
        WRITELN('NAME OF SOURCE FILE-->');
        GOTOXY(22,2);
        READLN(SOURCE);
        ERASE;
       END;
     5:BEGIN
         WHICH:=0;
         EXIT(TRANSONE);
       END;
     END;
  UNTIL WHICH IN [1..5];

 COUNTSTR:= CONCAT(SOURCE,'.COUNT');
 WRITELN;
 WRITELN('ENTER DRIVE NUMBER AND NAME');
 WRITELN('OF DESTINATION FILE-->');
 GOTOXY(22,2);
 READLN(DESTINATION);
 GOTOXY(42,1);
 DESTINATION:=CONCAT(DESTINATION,'.TEXT');  (* TAKE CARE OF .TEXT *)
 RESET(H,COUNTSTR);
 GET(H);
```

```
END;

PROCEDURE IOPUT;

(* THIS PROCEDURE PROVIDES INPUT FOR THE RECORD *)
(* NUMBERS AND INSURES THEY ARE WITHIN RANGE    *)


  BEGIN          (* BEGIN MAIN IOPUT *)
      GOTOXY(0,7);
      WRITE('ENTER START RECORD-->');
      GOTOXY(22,7);
      READLN(START);
      LINES(2);
      WRITE('ENTER STOP RECORD-->');
      GOTOXY(21,10);
      READLN(EXTENT);
      GOTOXY(42,1);
      EXTENT:=EXTENT+1;
      IF (START>EXTENT-1) THEN
       BEGIN
        ERASE;
        POKE(0,-16299);   (* TOGGLE PAGE 2 *)
        LINES(3);
        WRITELN('THE STARTING PAGE NUMBER CANNOT BE');
        WRITELN('GREATER THAN THE ENDING PAGE NUMBER');
        LINES(3);
        WRITELN('PRESS RETURN TO CONTINUE');
        INVLN(0,23,9);
        POKE(0,-16300);   (* TOGGLE PAGE 1 *)
        GOTOXY(42,0);
        READLN;
        ERASE;
       END;
      IF (EXTENT>H^) THEN
       BEGIN
        ERASE;
        POKE(0,-16299);   (* TOGGLE PAGE 2 *)
        LINES(3);
        WRITELN('THERE ARE ONLY ',H^,' RECORDS IN THE FILE');
        LINES(3);
        WRITELN('RECORD NUMBERING STARTS AT ZERO');
        LINES(3);
        WRITELN('PRESS RETURN TO CONTINUE');
        INVLN(0,23,12);
        POKE(0,-16300);   (* TOGGLE PAGE 1 *)
        GOTOXY(42,0);
        READLN;
        ERASE;
       END;
END;      (* END IOPUT           *)

BEGIN     (* BEGIN MAIN TRANSONE *)

  BREAKUP;
  ERASE;
  WRITELN;
```

```
    WRITELN('THERE ARE ',H^,' RECORDS');
    LINES(3);
    WRITE('SHOULD I TRANSFER ALL RECORDS?  Y/N');
    GOTOXY(36,5);
    RESET(L,SOURCE);
    REWRITE(G,DESTINATION);
    READ(ANSWER);

    IF ANSWER='Y' THEN
      BEGIN
        START:=0;
        EXTENT:=H^;
      END
    ELSE
      BEGIN
      REPEAT      (* REPEAT UNTIL USABLE INPUT *)
       IOPUT;
      UNTIL (START<EXTENT) AND (EXTENT<=H^) ;
      END;
    SEEK(L,START);
    START:=START+1;
        ERASE;
        LINES(2);
        WRITELN('THE DATA FROM RECORD # ');
        WRITELN('HAVE BEEN TRANSFERRED');
        GOTOXY(42,1);

    FOR COUNT:=START TO EXTENT DO
      BEGIN
        GET(L);
        WRITELN(G,'START');

        FOR I:=1 TO 20 DO
          BEGIN
            DUMMY:=L^[I];
            WRITELN(G,DUMMY);
          END;
        WRITELN(G,'END');
        GOTOXY(23,2);
        WRITELN(COUNT-1);
        GOTOXY(42,1);
       END;
    WRITELN(G,'FINISHED');
    WHICH:=0;              (* REMOVING WHICH:=0 WILL EXIT PROGRAM *)
    CLOSE(G,LOCK);
    CLOSE(L,LOCK);
    CLOSE(H,LOCK);
    ERASE;
  END; (* END TRANSONE *)

  (* THIS PROCEDURE IS USED TO MOVE DATA FROM A TEXT FILE *)
  (* INTO A FILE WITH THE FORMAT USED BY THE VIDEODISC    *)
  (* PROGRAM                                              *)

  SEGMENT PROCEDURE MOVERONE;

  PROCEDURE BREAKUP;
```

```
BEGIN
 REPEAT (* REPEAT UNTIL USABLE INPUT *)
   ERASE;
   GOTOXY(0,1);
   WRITELN('TRANSFER AN UNFORMATTED FILE TO A');
   WRITELN;
   WRITELN('FORMATTED FILE');
   GOTOXY(0,6);
   WRITELN('ENTER YOUR CHOICE-->');
   LINES(2);
   WRITELN('1. DESTINATION FILE TEXT.ONE ON DRIVE #5');
   WRITELN;
   WRITELN('2. DESTINATION FILE TEXT.TWO ON DRIVE #5');
   WRITELN;
   WRITELN('3. DESTINATION FILE INTRO    ON DRIVE #5');
   WRITELN;
   WRITELN('4. OTHER');
   WRITELN;
   WRITELN('5. RETURN TO MAIN MENU');
   GOTOXY(20,6);
   READLN(WHICH);
   GOTOXY(42,1);

     CASE WHICH OF

       1:DESTINATION:='#5:TEXT.ONE';
       2:DESTINATION:='#5:TEXT.TWO';
       3:DESTINATION:='#5:INTRO';
       4:BEGIN
          ERASE;
          WRITELN;
          WRITELN('ENTER NAME AND DRIVE NUMBER');
          WRITELN('OF DESTINATION FILE-->');
          GOTOXY(22,2);
          READLN(DESTINATION);
         END;
       5:BEGIN
          WHICH:=0;
          EXIT(MOVERONE);
         END;
     END;
    UNTIL WHICH IN [1..5];

    ERASE;
    GOTOXY(0,2);
    WRITELN('ENTER DRIVE NUMBER AND');
    WRITELN('NAME OF SOURCE FILE-->');
    GOTOXY(22,3);
    READLN(SOURCE);
    SOURCE:=CONCAT(SOURCE,'.TEXT');    (* TAKE CARE OF .TEXT *)

  END;  (* END BREAKUP          *)

  BEGIN (* BEGIN MOVERONE MAIN *)

  BREAKUP;
  ERASE;
  COUNTSTR:= CONCAT(DESTINATION,'.COUNT');
```

```
GOTOXY(0,2);
WRITELN('DOES THE DESTINATION EXIST? Y/N ');
GOTOXY(32,3);
READ(ANSWER);
RESET(G,SOURCE);
IF ANSWER<>'Y' THEN
  BEGIN
    REWRITE(H,COUNTSTR);
    CLOSE(H,LOCK);
    RESET(H,COUNTSTR);
    REWRITE(L,DESTINATION); (* CREATE FILES *)
    COUNT:=0;
    H^:=0;
  END
ELSE
  BEGIN
    ERASE;
    WRITELN;
    WRITELN('ENTER DESTINATION RECORD-->');
    GOTOXY(27,1);
    READLN(I);
    RESET(L,DESTINATION);   (* OPEN FILES *)
    RESET(H,COUNTSTR);
    GET(H);
    COUNT:= I;
  END;
SEEK(L,I);
ERASE;
GOTOXY(0,3);
WRITELN('DATA ARE BEING TRANSFERRED');
WRITELN;
WRITELN('THE DATA HAVE BEEN TRANSFERRED');
WRITELN('INTO RECORD #');
GOTOXY(42,1);
REPEAT
  (* FILL ALL THE ELEMENTS OF L *)
  FOR I:=1 TO 20 DO L^[I]:=' ';
  (* FIND THE START OF A SECTION OF TEXT *)
  (* OR THE END OF THE FILE              *)
  REPEAT
    READLN(G,DUMMY);
  UNTIL (DUMMY='START') OR (DUMMY='FINISHED');
  (* IF THIS IS THE START OF TEXT, TRANSFER THE TEXT TO L  *)
  (* STOP WHEN YOU HIT AN 'END' OR WHEN 20 LINES HAVE BEEN *)
  (* TRANSFERRED                                           *)
  IF DUMMY='START' THEN
    BEGIN
      I:=0;
      REPEAT
        I:=I+1;
        READLN(G,DUMMY);
        IF LENGTH(DUMMY)>40 THEN DUMMY:=COPY(DUMMY,1,40);
        L^[I]:=DUMMY;
      UNTIL (DUMMY='END') OR (I=20);
      IF L^[I]='END' THEN L^[I]:=' ';
      PUT(L);
      GOTOXY(13,6);
      WRITELN(COUNT);
```

```
        GOTOXY(42,1);
        COUNT:=COUNT+1;
      END;
  UNTIL DUMMY='FINISHED';
  IF COUNT>H^ THEN
    BEGIN
      H^:=COUNT;
      SEEK(H,0);
      (* UPDATE THE COUNTER FILE IF ANY NEW RECORDS HAVE BEEN ADDED *)
      PUT(H);
    END;
  WHICH:=0; (* REMOVE TO EXIT PROGRAM *)
  CLOSE(G,LOCK);
  CLOSE(L,LOCK);
  CLOSE(H,LOCK);
END; (* END MOVERONE *)


SEGMENT PROCEDURE MOVERTWO;

(* THIS PROCEDURE CONVERTS THE DATA FROM A TEXT FILE FORMAT *)
(* TO THE FORMAT USED BY PROGRAM                            *)


BEGIN
  ERASE;
  LINES(2);
  WRITELN('ENTER YOUR CHOICE-->');
  WRITELN;
  WRITELN('1. MOVE TEXT FILE INTO POINT.ONE');
  WRITELN;
  WRITELN('2. RETURN TO MAIN MENU');
  GOTOXY(20,2);
  READLN(WHICH);
  IF WHICH = 2 THEN
    BEGIN
     WHICH:=0; (* RETURN TO MAIN MENU *)
     EXIT(MOVERTWO);
    END;
  ERASE;
  GOTOXY(0,2);
  WRITELN('ENTER SOURCE DRIVE NUMBER');
  WRITELN('AND FILE NAME-->');
  GOTOXY(16,3);
  READLN(SOURCE);
  SOURCE:=CONCAT(SOURCE,'.TEXT');  (* TAKE CARE OF .TEXT *)
  WRITELN;
  WRITELN('IS DESTINATION #5:POINT.ONE? Y/N');
  GOTOXY(34,5);
  READ(ANS);
  GOTOXY(42,1);
  IF ANS = 'Y' THEN
    DESTINATION:='#5:POINT.ONE'
  ELSE
    BEGIN
      ERASE;
      WRITELN;
      WRITELN('ENTER DESTINATION DRIVE NUMBER');
```

```
      WRITELN('AND FILE NAME-->');
      GOTOXY(16,2);
      READLN(DESTINATION);
      GOTOXY(42,2);
    END;
  ERASE;
  WRITELN;
  WRITELN('ENTER DESTINATION RECORD-->');
  GOTOXY(27,1);
  READLN(COUNT);
  ERASE;
  WRITELN;
  WRITELN('DATA ARE BEING TRANSFERRED.');
  GOTOXY(42,2);
  RESET(F,DESTINATION);
  RESET(G,SOURCE);
  SEEK(F,COUNT);
  COUNTER:=COUNT;
  LINES(3);
  WRITELN('THE DATA HAVE BEEN TRANSFERRED ');
  WRITELN('INTO RECORD # ');
  GOTOXY(42,1);
  REPEAT
    (* FIND THE START OF A SECTION OF TEXT OR THE END OF THE FILE *)
    REPEAT
      READLN(G,DUMMY);
    UNTIL (DUMMY='START') OR (DUMMY='FINISHED');

  (* IF TEXT, CONVERT THE TEXT AND THEN TRANSFER IT                *)
  IF DUMMY='START' THEN
    BEGIN
      GET(F);
      I:=0;
      REPEAT
        I:=I+1;
        READLN(G,DUMMY);
        IF LENGTH(DUMMY)>40 THEN DUMMY:=COPY(DUMMY,1,40);
         (* ONLY TAKE 40 CHARACTERS                                 *)
        F^.ANSWERS[I]:=DUMMY;
      UNTIL (DUMMY='END') OR (I=20);
      IF DUMMY='END' THEN F^.ANSWERS[I]:=' ';
      (* MAKE SURE YOU PUT THE DATA BACK WHERE IT BELONGS *)
      SEEK(F,COUNTER);
      PUT(F);
      GOTOXY(14,6);
      WRITELN(COUNTER);
      GOTOXY(42,1);
      COUNTER:=COUNTER+1;
    END;

  UNTIL DUMMY='FINISHED';
  WHICH:=0; (* REMOVE TO EXIT PROGRAM                             *)
  CLOSE(F,LOCK);
  CLOSE(G,LOCK);
  ERASE;
END; (* END MOVERTWO *)
```

```
SEGMENT PROCEDURE TRANSTWO;

(* THIS SEGMENT CONVERTS THE FORMATTED STUDENT'S CHOICES *)
(* INTO A TEXT FILE FORMAT                               *)

PROCEDURE BREAKUP;

BEGIN
 ERASE;
 LINES(2);
 WRITELN('ENTER YOUR CHOICE-->');
 WRITELN;
 WRITELN('1. MOVE TEXT OF POINT.ONE TO TEXT FILE');
 WRITELN;
 WRITELN('2. RETURN TO MAIN MENU');
 GOTOXY(20,2);
 READLN(WHICH);
  IF WHICH = 2 THEN
   BEGIN
    WHICH:=0; (* REMOVE TO EXIT PROGRAM            *)
    EXIT(TRANSTWO);
   END;
END;              (* END BREAKUP *)


PROCEDURE IOPUT;   (* BEGIN IOPUT *)

BEGIN
      ERASE;
      LINES(2);
      WRITE('ENTER START RECORD-->');
      GOTOXY(21,2);
      READLN(START);
      LINES(2);
      WRITE('ENTER STOP RECORD-->');
      GOTOXY(20,5);
      READLN(EXTENT);
      GOTOXY(42,6);
      IF (START>EXTENT) THEN
        BEGIN
          ERASE;
          POKE(0,-16299);
          LINES(3);
          WRITELN('THE START NUMBER CANNOT BE GREATER');
          WRITELN('THAN THE STOP NUMBER');
          LINES(3);
          WRITELN('PRESS RETURN TO CONTINUE');
          INVLN(0,23,9);
          POKE(0,-16300);
          GOTOXY(42,0);
          READLN;
          ERASE;
        END;
      IF (EXTENT>H^-1) THEN
      BEGIN
       ERASE;
       POKE(0,-16299);
       LINES(3);
```

```
        WRITELN('THERE ARE ONLY ',H^,' RECORDS');
        LINES(3);
        WRITELN('RECORD NUMBERING STARTS AT ZERO');
        LINES(3);
        WRITELN('PRESS RETURN TO CONTINUE');
        INVLN(0,23,12);
        POKE(0,-16300);
        GOTOXY(42,0);
        READLN;
        ERASE;
      END;

END;  (* END IOPUT     *)

BEGIN (* MAIN TRANSTWO *)

  BREAKUP;
  ERASE;
  GOTOXY(0,2);
  WRITELN('IS SOURCE FILE #5:POINT.ONE? Y/N');
  GOTOXY(34,2);
  READ(ANS);

  IF ANS = 'Y' THEN
    SOURCE:='#5:POINT.ONE'
  ELSE
    BEGIN
      LINES(2);
      WRITELN('ENTER THE DRIVE NUMBER AND');
      WRITELN('SOURCE FILE NAME-->');
      GOTOXY(19,5);
      READLN(SOURCE);
      GOTOXY(42,2);
    END;

  ERASE;
  COUNTSTR:= CONCAT(SOURCE,'.COUNT'); (* TAKE CARE OF COUNTER FILE *)
  WRITELN;
  WRITELN('ENTER THE DESTINATION DRIVE NUMBER AND');
  WRITELN('FILE NAME-->');
  GOTOXY(12,2);
  READLN(DESTINATION);
  DESTINATION:=CONCAT(DESTINATION,'.TEXT');
  GOTOXY(42,3);
  RESET(H,COUNTSTR);
  GET(H);
  LINES(2);
  WRITELN('THERE ARE ',H^,' RECORDS');
  LINES(2);
  WRITE('SHOULD I TRANSFER ALL RECORDS? Y/N');
  GOTOXY(35,8);
  READ(ANSWER);
  GOTOXY(42,1);
  IF ANSWER='Y' THEN
    BEGIN
      START:=1;
      EXTENT:=(H^-1);
    END
```

```
    ELSE
      REPEAT
        IOPUT;
      UNTIL (START<=EXTENT) AND (EXTENT<=H^ - 1);
    RESET(F,SOURCE);
    SEEK(F,START);
    REWRITE(G,DESTINATION);
    ERASE;
    WRITELN;
    WRITELN('THE DATA ARE BEING TRANSFERRED');

    (* GET THE DATA AND TRANSFER IT TO G.  THE DATE ARE *)
    (* BRACKETED WITH A 'START' AND 'END'                *)

    LINES(3);
    WRITELN('THE DATA FROM RECORD # ');
    WRITELN('HAVE BEEN TRANSFERRED');
    GOTOXY(42,1);

    FOR COUNT:=START TO EXTENT DO
      BEGIN
        GET(F);
        WRITELN(G,'START');
        FOR I:=1 TO 20 DO
          BEGIN
            DUMMY:=F^.ANSWERS[I];
            WRITELN(G,DUMMY);
          END;
        WRITELN(G,'END');
      GOTOXY(22,5);
      WRITELN(COUNT);
      GOTOXY(42,1);

      END;
    WRITELN(G,'FINISHED');
    CLOSE(F,LOCK);
    CLOSE(G,LOCK);
    CLOSE(H,LOCK);
    WHICH:=0; (* REMOVE TO EXIT PROGRAM *)
    ERASE;
  END; (* END TRANSTWO *)
```

```
SEGMENT PROCEDURE FILL;

(* THIS SEGMENT ALLOWS THE FORMATTED ACCESS OF TEXT AND *)
(* POINT.ONE FILES                                      *)

VAR J,K:INTEGER;
    DEVICE:STRING;
    PTFILE,SPECIFIC,LIST:BOOLEAN;

(* THE BOOLEAN VARIABLE PTFILE SET PROGRAM FOR          *)
(* POINT.ONE TYPE OF FILE                               *)
(* SPECIFIC ALLOWS INPUT OF EXPLICIT INPUT OF STARTING  *)
(* AND STOPPING VALUES FOR RECORD RETRIVAL              *)
(* LIST SETS PRINTER                                    *)


PROCEDURE DASH; (* DRAWS DELIMITTERS IN PRINTED FILES   *)

BEGIN
  FOR I:=1 TO 40 DO
    BEGIN
      WRITE(G,'*');    (* G IS CONSOLE OR PRINTER *)
    END;
  WRITELN(G);          (* G IS CONSOLE OR PRINTER *)
END;

PROCEDURE IOPUT; (* THIS PROCEDURE INPUTS RECORD NUMBER *)

BEGIN
      ERASE;
      LINES(2);
      WRITELN('ENTER STARTING PAGE NUMBER-->');
      GOTOXY(29,2);
      READLN(START);
      WRITELN;
      WRITELN('ENTER ENDING PAGE NUMBER-->');
      GOTOXY(27,4);
      READLN(EXTENT);
      GOTOXY(42,0);
      IF (EXTENT > H^ - 1) THEN
        BEGIN
         ERASE;
         POKE(0,-16299);
         GOTOXY(0,3);
         WRITELN(' THERE ARE ONLY ',H^,' RECORDS IN THE FILE');
         LINES(3);
         WRITELN(' RECORD NUMBERING STARTS AT ZERO');
         LINES(3);
         WRITELN('  PRESS RETURN TO CONTINUE');
         INVLN(2,25,12);
         POKE(0,-16300);
         GOTOXY(42,0);
         READLN;
        END;

      IF (START > EXTENT) THEN
       BEGIN
        ERASE;
```

```
          POKE(0,-16299);
          LINES(3);
          WRITELN(' ENDING PAGE NUMBER CANNOT BE LESS THAN');
          WRITELN(' STARTING PAGE NUMBER');
          LINES(3);
          WRITELN(' PRESS RETURN TO CONTINUE');
          INVLN(1,24,9);
          POKE(0,-16300);
          GOTOXY(42,0);
          READLN;
         END;

END;   (* END IOPUT *)

PROCEDURE PTFILL;

(* PTFILL IS ACCESSED FOR THE POINT.ONE TYPE OF FILE *)

BEGIN
  REWRITE(G,DUMMY);       (* G IS CONSOLE OR PRINTER *)
  RESET(H,SOURCE);
  SEEK(H,0);
  GET(H);
  RESET(F,DEVICE);
  IF SPECIFIC THEN
    BEGIN
     REPEAT
      IOPUT;
     UNTIL (START<=EXTENT) AND (EXTENT<=H^ - 1);
    END
  ELSE
    BEGIN
     START:=0;
     EXTENT:=H^ - 1;
    END;
  FOR K:= START TO EXTENT DO
    BEGIN
      ERASE;
      IF LIST THEN DASH;
      WRITELN(G);
      WRITELN(G,'RECORD NUMBER: ',K);
      WRITELN(G);
      IF LIST THEN
        BEGIN
          DASH;
          WRITELN(G);
        END;
      IF NOT LIST THEN
        BEGIN
          WRITELN;
          WRITELN('PRESS RETURN TO CONTINUE');
          GOTOXY(42,0);
          READLN;
          ERASE;
        END;
      SEEK(F,K);
      GET(F);
      FOR J:=1 TO 20 DO
```

```
          BEGIN
           IF J<10 THEN
            WRITELN(G,J,'  ',F^.ANSWERS[J])  (* G IS CONSOLE OR PRINTER *)
           ELSE
            WRITELN(G,J,' ',F^.ANSWERS[J]);
          END;
        IF LIST THEN
          BEGIN
            DASH;
            FOR I:=1 TO 3 DO
              BEGIN
                WRITELN(G);                    (* G IS CONSOLE OR PRINTER *)
              END;
          END;
        IF NOT LIST THEN
          BEGIN
            REPEAT
              WRITE('ANY CHANGES TO BE MADE? Y/N ' );
              READ(ANSWER);
              WRITELN;
              IF ANSWER='Y' THEN
                BEGIN
                  WRITE('ENTER THE LINE NUMBER-->');
                  READLN(J);
                  WRITELN('ENTER NEW INFORMATION');
                  READLN(F^.ANSWERS[J]);
                  SEEK(F,K);
                  PUT(F);
                  ERASE;
                  FOR J:=1 TO 20 DO
                    BEGIN
                     IF J> 9 THEN
                      WRITELN(J,' ',F^.ANSWERS[J])
                     ELSE
                      WRITELN(J,'  ',F^.ANSWERS[J]);
                    END;
                END;
              UNTIL ANSWER<>'Y';
          END;
      END;
    ERASE;
    CLOSE(F,LOCK);
    CLOSE(H,LOCK);
    CLOSE(G);
    WHICH:=0;
  END;            (* END PTFILL *)


  PROCEDURE TXTFILL;

  (* THIS PROCEDURE ALLOWS ACCESS IN FORMATTED FORM TO *)
  (* TEXT.ONE TYPE FILES                               *)

  BEGIN
    REWRITE(G,DUMMY);       (* G IS CONSOLE OR PRINTER *)
    RESET(H,SOURCE);
    SEEK(H,0);
    GET(H);
```

```
RESET(L,DEVICE);
IF SPECIFIC THEN
  BEGIN
   REPEAT
     IOPUT;
   UNTIL (START<=EXTENT) AND (EXTENT<=H^ - 1);
  END
ELSE
  BEGIN
   START:=0;
   EXTENT:=H^ - 1;
  END;
FOR K:= START TO EXTENT DO
  BEGIN
    ERASE;
    IF LIST THEN DASH;
    WRITELN(G);
    WRITELN(G,'RECORD NUMBER: ',K);
    WRITELN(G);
    IF LIST THEN
     BEGIN
      DASH;
      WRITELN(G);
     END;
    IF NOT LIST THEN
      BEGIN
        WRITELN;
        WRITELN('PRESS RETURN TO CONTINUE');
        GOTOXY(42,0);
        READLN;
        ERASE;
      END;
    SEEK(L,K);
    GET(L);
    FOR J:=1 TO 20 DO
      BEGIN
       IF J<10 THEN
        WRITELN(G,J,'  ',L^[J])  (* G IS CONSOLE OR PRINTER *)
       ELSE
        WRITELN(G,J,' ',L^[J]);
      END;
    IF LIST THEN
      BEGIN
        DASH;
        FOR I:=1 TO 4 DO
          BEGIN
            WRITELN(G);          (* G IS CONSOLE OR PRINTER *)
          END;
      END;
    IF NOT LIST THEN
      BEGIN
        REPEAT
          WRITE('ANY CHANGES TO BE MADE? Y/N ' );
          READ(ANSWER);
          WRITELN;
          IF ANSWER='Y' THEN
            BEGIN
              WRITE('ENTER THE LINE NUMBER-->');
```

```
                    READLN(J);
                    WRITELN('ENTER NEW INFORMATION');
                    READLN(L^[J]);
                    SEEK(L,K);
                    PUT(L);
                    ERASE;
                    FOR J:=1 TO 20 DO
                      BEGIN
                       IF J<10 THEN
                        WRITELN(J,'  ',L^[J])
                       ELSE
                        WRITELN(J,' ',L^[J]);
                      END;
                  END;
                UNTIL ANSWER<>'Y';
            END;
        END;
      ERASE;
      CLOSE(L,LOCK);
      CLOSE(H,LOCK);
      CLOSE(G);
      WHICH:=0;
    END;            (* END TXTFILL *)

    PROCEDURE HOW;

    (* THIS PROCEDURE DETERMINES HOW TO TREAT THE FILES *)

    BEGIN           (* START HOW *)
    REPEAT
      ERASE;
      LINES(2);
      WRITELN('1. REVIEW AN ENTIRE FILE?');
      WRITELN;
      WRITELN('2. REVIEW PART OF A FILE?');
      WRITELN;
      WRITELN('3. PRINT OUT AN ENTIRE FILE?');
      WRITELN;
      WRITELN('4. PRINT OUT PART OF A FILE?');
      LINES(2);
      WRITE('ENTER YOUR CHOICE-->');
      GOTOXY(20,11);
      READLN(COUNT);
      GOTOXY(42,0);

      CASE COUNT OF

          1:DUMMY:='CONSOLE:';
          2:BEGIN
              SPECIFIC:=TRUE;
              DUMMY:='CONSOLE:';
            END;
          3:BEGIN
              LIST:=TRUE;
              DUMMY:='PRINTER:';
            END;
          4:BEGIN
              LIST:=TRUE;
```

```
            SPECIFIC:=TRUE;
            DUMMY:='PRINTER:';
          END;
      END;
    UNTIL COUNT IN [1..4];
END;              (* END HOW    *)

BEGIN             (* BEGIN MAIN *)
REPEAT
  REPEAT
    NEW:=FALSE; (* INITIALIZE *)
    ADD:=FALSE;
    LIST:=FALSE;
    PTFILE:=FALSE;
    SPECIFIC:=FALSE;
    ERASE;
    LINES(2);
    WRITELN('VIEW FILES IN FORMATTED FORM');
    LINES(2);
    WRITELN('ENTER YOUR CHOICE-->');
    WRITELN;
    WRITELN('1. SOURCE FILE TEXT.ONE ON DRIVE #5');
    WRITELN;
    WRITELN('2. SOURCE FILE TEXT.TWO ON DRIVE #5');
    WRITELN;
    WRITELN('3. SOURCE FILE INTRO    ON DRIVE #5');
    WRITELN;
    WRITELN('4. SOURCE FILE POINT.ONE ON DRIVE #5');
    WRITELN;
    WRITELN('5. OTHER');
    WRITELN;
    WRITELN('6. RETURN TO MAIN MENU');
    GOTOXY(20,5);
    READLN(WHICH);
    GOTOXY(42,0);

    CASE WHICH OF

     1:DEVICE:='#5:TEXT.ONE';
     2:DEVICE:='#5:TEXT.TWO';
     3:DEVICE:='#5:INTRO';
     4:BEGIN
        DEVICE:='#5:POINT.ONE';
        PTFILE:=TRUE;
       END;
     5:BEGIN
        ERASE;
        LINES(2);
        WRITELN('ENTER SOURCE FILE DRIVE AND');
        WRITELN('NAME-->');
        GOTOXY(7,3);
        READLN(DEVICE);
        GOTOXY(42,3);
        LINES(2);
        WRITELN('IS THIS A POINT.ONE TYPE FILE? Y/N');
        GOTOXY(34,5);
        READ(ANSWER);
        GOTOXY(42,6);
```

```
        IF ANSWER = 'Y' THEN PTFILE := TRUE;
        END;
      6:BEGIN
        WHICH:=0;
        EXIT(FILL);
        END;
     END;
   UNTIL WHICH IN [1..6];

    SOURCE:=CONCAT(DEVICE,'.COUNT');
     HOW;
     IF PTFILE THEN
      PTFILL
     ELSE
      TXTFILL;
    UNTIL WHICH IN [1..6];
    WHICH:=0;                    (* WHICH SETS RETURN TO MAIN MENU *)
END;

SEGMENT PROCEDURE CHECK; (* BEGIN CHECK *)

(* THIS SEGMENT PROVIDES FOR READING, REPLACING, OR CREATING *)
(* COUNTER FILES                                             *)

PROCEDURE BREAKUP;

BEGIN
  REPEAT
       ERASE;
       GOTOXY(0,2);
       WRITELN('CHECK COUNTER FILES');
       LINES(2);
       WRITELN('ENTER YOUR CHOICE-->');
       LINES(2);
       WRITELN('1. SOURCE TEXT.ONE.COUNT ON DRIVE #5');
       WRITELN;
       WRITELN('2. SOURCE TEXT.TWO.COUNT ON DRIVE #5');
       WRITELN;
       WRITELN('3. SOURCE INTRO.COUNT    ON DRIVE #5');
       WRITELN;
       WRITELN('4. SOURCE POINT.ONE.COUNT ON DRIVE #5');
       WRITELN;
       WRITELN('5. OTHER');
       WRITELN;
       WRITELN('6. RETURN TO MAIN MENU');
       GOTOXY(20,5);
       READLN(WHICH);
       GOTOXY(42,4);
  UNTIL WHICH IN [1..6];

  CASE WHICH OF

    1:SOURCE:='#5:TEXT.ONE.COUNT';
    2:SOURCE:='#5:TEXT.TWO.COUNT';
    3:SOURCE:='#5:INTRO.COUNT';
    4:SOURCE:='#5:POINT.ONE.COUNT';
    5:BEGIN
        ERASE;
```

```
            GOTOXY(0,6);
            WRITELN('ENTER SOURCE DRIVE NUMBER AND ');
            WRITELN('FILE NAME-->');
            GOTOXY(12,7);
            READLN(SOURCE);
            GOTOXY(42,2);
          END;
        6:BEGIN
            WHICH:=0; (* REMOVE TO EXIT PROGRAM *)
            EXIT(CHECK);
          END;
      END;

END;  (* END BREAKUP       *)

BEGIN (* BEGIN MAIN CHECK *)
  BREAKUP;
  ERASE;
  LINES(2);
  WRITELN('SHOULD I CREATE A NEW FILE? Y/N');
  GOTOXY(31,2);
  READ(ANSWER);
  GOTOXY(42,5);
  IF ANSWER='Y' THEN
    BEGIN
      REWRITE(H,SOURCE);
      SEEK(H,0);
      WRITELN;
      WRITELN('ENTER THE NEW VALUE-->');
      GOTOXY(22,6);
      READLN(H^);
      GOTOXY(42,5);
      PUT(H);
    END
  ELSE
    BEGIN
      ERASE;
      RESET(H,SOURCE);
      SEEK(H,0);
      GET(H);
      WRITELN('THE VALUE IS ',H^);
      LINES(2);
      WRITE('SHOULD I CHANGE THE VALUE?  Y/N');
      GOTOXY(31,3);
      READ(ANSWER);
      GOTOXY(42,3);
      IF ANSWER='Y' THEN
        BEGIN
          LINES(2);
          WRITE('ENTER THE NEW VALUE-->');
          GOTOXY(22,5);
          READLN(H^);
          GOTOXY(42,6);
          SEEK(H,0);
          PUT(H);
        END;
    END;
  ERASE;
```

```
  WHICH:=0; (* WHICH ALSO SETS RETURN TO MENU *)
  CLOSE(H,LOCK);
END;(*END CHECK*)


SEGMENT PROCEDURE TRANSFER;

BEGIN (* MAIN *)
 REPEAT
   ERASE;
   WRITELN;
   WRITELN('    TRANSFER MAIN MENU');
   LINES(2);
   WRITELN('ENTER YOUR CHOICE-->');
   LINES(2);
   WRITELN('1. MOVE A FORMATTED FILE TO A TEXT FILE');
   WRITELN;
   WRITELN('2. MOVE A TEXT FILE TO A FORMATTED FILE');
   WRITELN;
   WRITELN('3. MOVE POINT.ONE TO A TEXT FILE');
   WRITELN;
   WRITELN('4. MOVE A TEXT FILE TO POINT.ONE ');
   WRITELN;
   WRITELN('5. REVIEW FILES IN FORMATTED FORM');
   WRITELN;
   WRITELN('6. CHECK COUNTER FILE');
   WRITELN;
   WRITELN('7. EXIT PROGRAM');

   GOTOXY(20,4);
   READLN(WHICH);
   GOTOXY(42,2);

   CASE WHICH OF

     1:TRANSONE;
     2:MOVERONE;
     3:TRANSTWO;
     4:MOVERTWO;
     5:FILL;
     6:CHECK;
     7:BEGIN
        ERASE;
        EXIT(TRANSFER);
       END;
     END;
  UNTIL WHICH = 7;  (* LOOP UNTIL CORRECT *)
END;
```

```
SEGMENT PROCEDURE STRUCTURE;
BEGIN
  REPEAT
    FIRST;
    IF NOT LEAVE THEN
      BEGIN
        REPEAT
          ERASE;
          WRITELN(OUT);
          WRITELN(OUT,'RECORD NUMBER: ',H^);
          WRITELN(OUT);
          IF NOT LIST THEN
            BEGIN
              WRITE('PRESS RETURN TO CONTINUE');
              READLN;
            END;
          GET(F);
          IF EVERY THEN
            BEGIN
              ORIGIN:=1;
              EXTENT:=8;
            END
          ELSE
            BEGIN
              ORIGIN:=VALUE;
              EXTENT:=VALUE;
            END;
          FOR COUNTER:=ORIGIN TO EXTENT DO
            BEGIN
              SECOND;
            END;
          SEEK(F,H^);
          PUT(F);
          H^:=H^+1;
          IF (NOT NEW) AND (NOT ADD) THEN ANSWER:='N'
          ELSE
            BEGIN
              WRITELN;
              WRITE('FINISHED? ');
              READLN(RESPONSE);
              ERASE;
            END;
        UNTIL (RESPONSE='Y') OR (H^=STOP);
        IF NEW OR ADD THEN
          BEGIN
            SEEK(H,0);
            PUT(H);
          END;
        CLOSE(F,LOCK);
        CLOSE(H,LOCK);
        CLOSE(OUT);
      END;
  UNTIL COUNT=7;
END;
```

```
SEGMENT PROCEDURE SECOND;

PROCEDURE DUMMI;
BEGIN
  WITH F^ DO
    BEGIN
      FOR I:=1 TO 5 DO
        BEGIN
          SEQMANY[I]:=1;
          FEELMANY[I]:=1;
          FEEDWHERE[I]:=1;
          NEXT[I]:=1;
          DONE[I]:=FALSE;
          FOR J:=1 TO 2 DO
            BEGIN
              SEQ2[I,J]:='1';
              SEQ3[I,J]:='1';
            END;
          FOR J:=3 TO 4 DO
            BEGIN
              SEQ3[I,J]:='1';
            END;
        END;
    END;
END;

PROCEDURE STARTA;
BEGIN
  ERASE;
  CHANGE:=TRUE;
  IF LIST THEN WRITELN(OUT);
END;

PROCEDURE EH;
BEGIN
  READLN(RESPONSE);
  IF RESPONSE='Y' THEN FEEDBACK:=TRUE
  ELSE FEEDBACK:=FALSE;
END;

PROCEDURE CHANGEIT;
BEGIN
  WRITELN(OUT);
  IF LIST THEN CHANGE:=FALSE
  ELSE
    BEGIN
      WRITE('MAKE CHANGES ? ');
      EH;
      CHANGE:=FEEDBACK;
      IF FEEDBACK THEN ERASE;
    END;
END;

PROCEDURE WHICH;
BEGIN
  IF FEEDBACK THEN WRITELN(OUT,' TRUE ')
  ELSE WRITELN(OUT,' FALSE ');
END;
```

```
            IF I<10 THEN WRITELN(OUT,I,'  ',ANSWERS[I])
            ELSE WRITELN(OUT,I,' ',ANSWERS[I]);
          END;
        WRITELN(OUT);
        IF LIST THEN CHANGE:=FALSE
        ELSE
          BEGIN
            WRITE('MAKE CHANGES ? ');
            EH;
            WIPE;
            CHANGE:=FEEDBACK;
          END;
        IF CHANGE THEN
          BEGIN
            WRITE('ENTER LINE NUMBER ');
            READLN(I);
            WIPE;
            WRITELN('ENTER CORRECT INFORMATION');
            READLN(ANSWERS[I]);
            ERASE;
          END
        ELSE
          BEGIN
            FINISHED:=TRUE;
            WRITELN(OUT);
          END;
      UNTIL FINISHED;
    END;
END;

BEGIN
  WITH F^ DO
    BEGIN
      STARTA;
      IF NEW OR ADD THEN
        BEGIN
          FOR I:=1 TO 20 DO
            BEGIN
              ANSWERS[I]:=' ';
            END;
        END
      ELSE
        BEGIN
          IF LOOK THEN SUB;
          IF CHANGE THEN
            BEGIN
              WRITELN('ENTER A SCREEN FULL OF ANSWERS');
              FOR I:=1 TO 20 DO
                BEGIN
                  IF I<10 THEN WRITELN(I,'  ')
                  ELSE WRITELN(I,' ');
                END;
              FOR I:=1 TO 20 DO
                BEGIN
                  GOTOXY(3,I);
                  READLN(ANSWERS[I]);
                END;
              STARTA;
```

```
                SUB;
              END;
          END;
      END;
END;

PROCEDURE THREE;
VAR ORIGIN, FAR:INTEGER;

PROCEDURE SUB;
BEGIN
  WITH F^ DO
    BEGIN
      FOR I:=1 TO 5 DO
        BEGIN
          WRITELN(OUT);
          WRITELN(OUT,'ANSWER #: ',I);
          FOR J:=1 TO 2 DO
            BEGIN
              WRITE(OUT,PROMPT[J]);
              WRITELN(OUT,POSITIONS[I,J]);
            END;
        END;
      FOR I:=1 TO 20 DO
        BEGIN
          GOTOXY(19,I);
          IF I<10 THEN WRITE(I,'  ',ANSWERS[I])
          ELSE WRITE(I,' ',ANSWERS[I]);
        END;
      CHANGEIT;
    END;
END;

BEGIN
  WITH F^ DO
    BEGIN
      STARTA;
      IF NEW OR ADD THEN
        BEGIN
          FOR I:=1 TO 5 DO
            BEGIN
              FOR J:=1 TO 2 DO
                BEGIN
                  POSITIONS[I,J]:=40;
                END;
            END;
        END
      ELSE
        BEGIN
          IF LOOK THEN SUB;
          IF CHANGE THEN
            BEGIN
              REPEAT
                ERASE;
                WRITELN('ENTER THE DEFINING LINE NUMBERS');
                WRITELN;
                WRITELN('FOR THE ANSWERS');
                WRITELN;
```

```
            WRITELN('DO YOU WISH TO ENTER LINE NUMBERS');
            WRITELN;
            WRITE('FOR ALL QUESTIONS? ');
            EH;
            IF FEEDBACK THEN ANSWER:='Y'
            ELSE ANSWER:='N';
            IF ANSWER='Y' THEN
              BEGIN
                ORIGIN:=1;
                FAR:=5;
              END
            ELSE
              BEGIN
                GOTOXY(0,9);
                WRITELN('ENTER START NUMBER ');
                WRITELN;
                WRITE('ENTER STOP NUMBER  ');
                GOTOXY(19,9);
                READLN(ORIGIN);
                GOTOXY(19,11);
                READLN(FAR);
              END;
            ERASE;
            GOTOXY(0,2);
            WRITE('ENTER LINE NUMBERS');
            WRITELN;
            WRITE('FOR ANSWERS');
            GOTOXY(0,6);
            WRITE('ANSWER NUMBER ');
            GOTOXY(0,8);
            WRITE(PROMPT[1]);
            GOTOXY(0,10);
            WRITE(PROMPT[2]);
            FOR I:=1 TO 20 DO
              BEGIN
                GOTOXY(19,I);
                IF I<10 THEN WRITE(I,'  ',ANSWERS[I])
                ELSE WRITE(I,' ',ANSWERS[I]);
              END;
            FOR I:=ORIGIN TO FAR DO
              BEGIN
                GOTOXY(14,6);
                WRITE(I);
                GOTOXY(14,8);
                WRITE('  ');
                GOTOXY(14,10);
                WRITE('  ');
                FOR J:=1 TO 2 DO
                  BEGIN
                    GOTOXY(14,(6+(2*J)));
                    READLN(POSITIONS[I,J]);
                  END;
              END;
            STARTA;
            SUB;
          UNTIL NOT CHANGE;
        END;
    END;
```

```
    END;
END;

PROCEDURE FOUR;

PROCEDURE SUB;
BEGIN
  WITH F^ DO
    BEGIN
      GOTOXY(0,2);
      WRITELN(OUT,'CORRECT CHOICE = ',CORRECT);
      CHANGEIT;
    END;
END;

BEGIN
  WITH F^ DO
    BEGIN
      STARTA;
      IF LOOK THEN SUB;
      IF CHANGE THEN
        BEGIN
          REPEAT
            GOTOXY(0,2);
            WRITE('ENTER THE CORRECT CHOICE NUMBER  ');
            READLN(CORRECT);
            STARTA;
            SUB;
          UNTIL NOT CHANGE;
        END;
    END;
END;

PROCEDURE FIVE;

PROCEDURE SUB;
BEGIN
  WITH F^ DO
    BEGIN
      GOTOXY(0,2);
      WRITE(OUT,'TEXT BEFORE SEQUENCE ONE (PED) = ');
      FEEDBACK:=TEXT[1,1];
      WHICH;
      WRITELN;
      WRITE(OUT,'TEXT BEFORE SEQUENCE ONE (EX)  = ');
      FEEDBACK:=TEXT[2,1];
      WHICH;
      WRITELN;
      WRITE(OUT,'TEXT AFTER SEQUENCE ONE  (PED) = ');
      FEEDBACK:=TEXT[1,2];
      WHICH;
      WRITELN;
      WRITE(OUT,'TEXT AFTER SEQUENCE ONE  (EX)  = ');
      FEEDBACK:=TEXT[2,2];
      WHICH;
      WRITELN;
      WRITE(OUT,'PLAY SEQUENCE ONE        (PED) = ');
      FEEDBACK:=PLAY[1,1];
```

```
        WHICH;
        WRITELN;
        WRITE(OUT,'PLAY SEQUENCE ONE        (EX)  = ');
        FEEDBACK:=PLAY[2,1];
        WHICH;
        WRITELN;
        CHANGEIT;
      END;
  END;

  BEGIN
    WITH F^ DO
      BEGIN
        STARTA;
        IF LOOK THEN SUB;
        IF CHANGE THEN
          BEGIN
            REPEAT
              GOTOXY(0,2);
              WRITELN('ANSWER WITH A Y<CR> OR A <CR> ');
              GOTOXY(0,5);
              WRITELN('TEXT BEFORE SEQUENCE ONE? (PED) ');
              WRITELN;
              WRITELN('TEXT BEFORE SEQUENCE ONE? (EX)  ');
              WRITELN;
              WRITELN('TEXT FOLLOWING SEQUENCE?  (PED) ');
              WRITELN;
              WRITELN('TEXT FOLLOWING SEQUENCE?  (EX)  ');
              WRITELN;
              WRITELN('PLAY A SEQUENCE?          (PED) ');
              WRITELN;
              WRITELN('PLAY A SEQUENCE?          (EX)  ');
              GOTOXY(34,5);
              EH;
              TEXT[1,1]:=FEEDBACK;
              GOTOXY(34,7);
              EH;
              TEXT[2,1]:=FEEDBACK;
              GOTOXY(34,9);
              EH;
              TEXT[1,2]:=FEEDBACK;
              GOTOXY(34,11);
              EH;
              TEXT[2,2]:=FEEDBACK;
              GOTOXY(34,13);
              EH;
              PLAY[1,1]:=FEEDBACK;
              GOTOXY(34,15);
              EH;
              PLAY[2,1]:=FEEDBACK;
              STARTA;
              SUB;
            UNTIL NOT CHANGE;
          END;
      END;
  END;

  PROCEDURE SIX;
```

```
PROCEDURE SUB;
BEGIN
  WITH F^ DO
    BEGIN
      GOTOXY(0,2);
      WRITELN(OUT,'NUMBER OF PAGES PRIOR = ',TEXTMANY[1]);
      WRITELN;
      WRITELN(OUT,'NUMBER OF PAGES AFTER = ',TEXTMANY[2]);
      WRITELN;
      WRITELN(OUT,'THEY START AT LOCATION  ',TEXTWHERE);
      CHANGEIT;
    END;
END;

BEGIN
  WITH F^ DO
    BEGIN
      STARTA;
      IF LOOK THEN SUB;
      IF CHANGE THEN
        BEGIN
          REPEAT
            GOTOXY(0,2);
            WRITELN('NUMBER OF PAGES PRIOR?       ');
            WRITELN;
            WRITELN('NUMBER OF PAGES FOLLOWING?   ');
            WRITELN;
            WRITELN('WHERE ON DISK DOES IT START? ');
            GOTOXY(32,2);
            READLN(TEXTMANY[1]);
            GOTOXY(32,4);
            READLN(TEXTMANY[2]);
            GOTOXY(32,6);
            READLN(TEXTWHERE);
            STARTA;
            SUB;
          UNTIL NOT CHANGE;
        END;
    END;
END;



PROCEDURE SEVEN;

PROCEDURE SUB;
BEGIN
  WITH F^ DO
    BEGIN
      GOTOXY(0,2);
      WRITE(OUT,'PICK UP A RESPONSE        = ');
      FEEDBACK:=RESPONSE;
      WHICH;
      WRITELN;
      WRITE(OUT,'PLAY SEQUENCE THREE (PED) = ');
      FEEDBACK:=PLAY[3,2];
      WHICH;
```

```
          WRITELN;
          WRITE(OUT,'PLAY SEQUENCE THREE (EX)  = ');
          FEEDBACK:=PLAY[2,2];
          WHICH;
          WRITELN;
          CHANGEIT;
        END;
    END;

    BEGIN
      WITH F^ DO
        BEGIN
          STARTA;
          IF NEW OR ADD THEN
            BEGIN
              RESPONSE:=TRUE;
              PLAY[1,2]:=TRUE;
              PLAY[2,2]:=TRUE;
            END
          ELSE
            BEGIN
              IF LOOK THEN SUB;
              IF CHANGE THEN
                BEGIN
                  REPEAT
                    GOTOXY(0,2);
                    WRITELN('ANSWER WITH A Y<CR> OR A <CR>');
                    GOTOXY(0,5);
                    WRITELN('IS THERE A RESPONSE?       ');
                    WRITELN;
                    WRITELN('PLAY SEQUENCE THREE? (PED) ');
                    WRITELN;
                    WRITELN('PLAY SEQUENCE THREE? (EX)  ');
                    GOTOXY(30,5);
                    EH;
                    RESPONSE:=FEEDBACK;
                    GOTOXY(30,7);
                    EH;
                    PLAY[1,2]:=FEEDBACK;
                    GOTOXY(30,9);
                    EH;
                    PLAY[2,2]:=FEEDBACK;
                    STARTA;
                    SUB;
                  UNTIL NOT CHANGE;
                END;
            END;
        END;
    END;

    PROCEDURE TITLE;
    BEGIN
      WRITELN;
      WRITELN(OUT,'ANSWER ',I);
      WRITELN;
    END;

    PROCEDURE ONEA;
```

```
PROCEDURE SUB;
BEGIN
  WITH F^ DO
    BEGIN
      WRITELN(OUT,'VALUES FOR SEQ2');
      WRITELN(OUT);
      FOR J:=1 TO 2 DO
        BEGIN
          WRITE(OUT,PROMPT[J]);
          WRITELN(OUT,SEQ2[I,J]);
          WRITELN(OUT);
        END;
      CHANGEIT;
      IF FEEDBACK THEN TITLE;
    END;
END;

BEGIN
  WITH F^ DO
    BEGIN
      STARTA;
      TITLE;
      IF LOOK THEN SUB;
      IF CHANGE THEN
        BEGIN
          REPEAT
            GOTOXY(0,3);
            WRITELN('ENTER THE START & STOP VALUES FOR SEQ2');
            WRITELN;
            FOR J:=1 TO 2 DO
              BEGIN
                WRITELN(PROMPT[J]);
                WRITELN;
              END;
            GOTOXY(14,5);
            READLN(SEQ2[I,1]);
            GOTOXY(14,7);
            READLN(SEQ2[I,2]);
            STARTA;
            TITLE;
            SUB;
          UNTIL NOT CHANGE;
        END;
    END;
END;

PROCEDURE TWOA;

PROCEDURE SUB;
BEGIN
  WITH F^ DO
    BEGIN
      WRITELN(OUT,'VALUES FOR SEQ3');
      WRITELN(OUT);
      FOR J:=1 TO 2 DO
        BEGIN
          WRITE(OUT,PROMPT[J]);
```

```
            WRITELN(OUT,SEQ3[I,J]);
            WRITELN(OUT);
          END;
        WRITELN(OUT);
        FOR J:=1 TO 2 DO
          BEGIN
            WRITE(OUT,PROMPT[J]);
            WRITELN(OUT,SEQ3[I,(J+2)]);
            WRITELN(OUT);
          END;
        WRITELN(OUT);
        WRITELN(OUT,'THERE ARE ',SEQMANY[I],' SEQUENCES');
        CHANGEIT;
        IF FEEDBACK THEN TITLE;
      END;
  END;

  BEGIN
    WITH F^ DO
      BEGIN
        STARTA;
        TITLE;
        IF LOOK THEN SUB;
        IF CHANGE THEN
          BEGIN
            REPEAT
              GOTOXY(0,3);
              WRITELN('ENTER THE FIRST SET OF');
              WRITELN;
              WRITELN('START & STOP VALUES FOR SEQ3');
              WRITELN;
              FOR J:=1 TO 2 DO
                BEGIN
                  WRITELN(PROMPT[J]);
                  WRITELN;
                END;
              WRITELN;
              WRITELN('ENTER THE SECOND SET OF VALUES');
              WRITELN;
              FOR J:=1 TO 2 DO
                BEGIN
                  WRITELN(PROMPT[J]);
                  WRITELN;
                END;
              WRITELN;
              WRITE('ENTER THE NUMBER OF SEQUENCES ');
              FOR J:=1 TO 4 DO
                BEGIN
                  CASE J OF
                    1:GOTOXY(14,7);
                    2:GOTOXY(14,9);
                    3:GOTOXY(14,14);
                    4:GOTOXY(14,16);
                  END;
                  READLN(SEQ3[I,J]);
                END;
              GOTOXY(32,19);
              READLN(SEQMANY[I]);
```

```
            STARTA;
            TITLE;
            SUB;
          UNTIL NOT CHANGE;
        END;
    END;
END;

PROCEDURE THREEA;

PROCEDURE SUB;
BEGIN
  WITH F^ DO
    BEGIN
      WRITELN(OUT,'THERE ARE ',FEEDMANY[I],' PAGES OF EXPLANATION');
      WRITELN;
      WRITELN(OUT,'THEY START AT LOCATION ',FEEDWHERE[I]);
      CHANGEIT;
      IF FEEDBACK THEN TITLE;
    END;
END;

BEGIN
  WITH F^ DO
    BEGIN
      STARTA;
      TITLE;
      IF LOOK THEN SUB;
      IF CHANGE THEN
        BEGIN
          REPEAT
            GOTOXY(0,3);
            WRITELN('HOW MANY PAGES OF EXPLANATION?    ');
            WRITELN;
            WRITELN('WHERE ON THIS DISK DOES IT START? ');
            GOTOXY(35,3);
            READLN(FEEDMANY[I]);
            GOTOXY(35,5);
            READLN(FEEDWHERE[I]);
            STARTA;
            TITLE;
            SUB;
          UNTIL NOT CHANGE;
        END;
    END;
END;

PROCEDURE FOURA;

PROCEDURE SUB;
BEGIN
  WITH F^ DO
    BEGIN
      WRITELN(OUT,'THE NEXT JUNCTION POINT IS AT RECORD ',NEXT[I]);
      WRITELN;
      WRITE(OUT,'THIS IS AN END POINT: ');
      FEEDBACK:=DONE[I];
      WHICH;
```

```
      CHANGEIT;
      IF FEEDBACK THEN TITLE;
    END;
END;

BEGIN
  WITH F^ DO
    BEGIN
      STARTA;
      TITLE;
      IF LOOK THEN SUB;
      IF CHANGE THEN
        BEGIN
          REPEAT
            GOTOXY(0,3);
            WRITELN('ENTER THE NEXT JUNCTION POINT ');
            WRITELN;
            WRITELN('IS THIS AN END POINT?         ');
            GOTOXY(32,3);
            READLN(NEXT[I]);
            GOTOXY(32,5);
            EH;
            DONE[I]:=FEEDBACK;
            STARTA;
            TITLE;
            SUB;
          UNTIL NOT CHANGE;
        END;
    END;
END;

PROCEDURE GIZMO(DUMMY,MIN,MAX:INTEGER);
BEGIN
  IF NOT (DUMMY IN [MIN..MAX]) THEN
    BEGIN
      WIPE;
      WRITELN('PLEASE ENTER A NUMBER BETWEEN ',MIN,' AND ',MAX);
      WRITELN;
      WRITE('PRESS <CR> TO CONTINUE');
      READLN;
      WIPE;
      WIPE;
      WIPE;
    END;
END;

PROCEDURE EIGHT;

VAR ORIGIN, ORIGIN1, EXTENT, EXTENT1, COUNT:INTEGER;

BEGIN
  ORIGIN:=1;
  EXTENT:=5;
  ORIGIN1:=1;
  EXTENT1:=4;
  IF (NOT NEW) AND (NOT ADD) THEN
    BEGIN
      ERASE;
```

```
        GOTOXY(0,2);
        WRITE('GO THROUGH ALL ANSWERS? ');
        EH;
        WRITELN;
        IF NOT FEEDBACK THEN
          BEGIN
            REPEAT
              WRITE('ENTER START ANSWER ');
              READLN(ORIGIN);
              GIZMO(ORIGIN,1,5);
            UNTIL ORIGIN IN [1..5];
            WRITELN;
            REPEAT
              WRITE('ENTER STOP NUMBER ');
              READLN(EXTENT);
              GIZMO(EXTENT,ORIGIN,5);
            UNTIL EXTENT IN [ORIGIN..5];
          END;
        WRITELN;
        WRITE('GO THROUGH ALL PROCEDURES? ');
        EH;
        IF NOT FEEDBACK THEN
          BEGIN
            WRITELN;
            WRITELN('1. SEQUENCE TWO');
            WRITELN;
            WRITELN('2. SEQUENCE THREE');
            WRITELN;
            WRITELN('3. PAGES OF EXPLANATION');
            WRITELN;
            WRITELN('4. NEXT JUNCTION POINT');
            WRITELN;
            REPEAT
              WRITE('ENTER THE PROCEDURE NUMBER ');
              READLN(ORIGIN1);
              GIZMO(ORIGIN1,1,4);
            UNTIL ORIGIN1 IN [1..4];
            EXTENT1:=ORIGIN1;
            ERASE;
          END;
      END;
    FOR I:=ORIGIN TO EXTENT DO
      BEGIN
        FOR COUNT:=ORIGIN1 TO EXTENT1 DO
          BEGIN
            CASE COUNT OF
              1:ONEA;
              2:TWOA;
              3:THREEA;
              4:FOURA;
            END;
          END;
      END;
  END;

  BEGIN
    CASE COUNTER OF
      1:ONE;
```

```
        2:TWO;
        3:THREE;
        4:FOUR;
        5:FIVE;
        6:SIX;
        7:SEVEN;
        8:BEGIN
            IF NEW OR ADD THEN
              BEGIN
                WRITELN;
                WRITE('SHOULD I DUMMY THE DATA? ');
                EH;
                IF FEEDBACK THEN DUMMI
                ELSE EIGHT;
              END
            ELSE EIGHT;
          END;
    END;
END;
```

```
SEGMENT PROCEDURE FIRST;

PROCEDURE FIND;
BEGIN
  CASE COUNT OF

    1:BEGIN
        GETCVAL(RESPONSE);
        IF RESPONSE<>'RUN' THEN
          BEGIN
            RESPONSE:='KRUNCHED';
            SETCVAL(RESPONSE);
            SETCHAIN('EXEC/KRUNCH');
            GOTOXY(42,10);
            WRITELN('PLEASE WAIT, PROCESSING');
            POKE(0,-16299);
            EXIT(PROGRAM);
          END
        ELSE NEW:=TRUE;
      END;

    2:LOOK:=TRUE;

    3:BEGIN
        LOOK:=TRUE;
        SPECIFIC:=TRUE;
      END;

    4:BEGIN
        LOOK:=TRUE;
        LIST:=TRUE;
      END;

    5:BEGIN
        LOOK:=TRUE;
        LIST:=TRUE;
        SPECIFIC:=TRUE;
      END;

    6:BEGIN
        GETCVAL(RESPONSE);
        IF RESPONSE<>'RUN' THEN
          BEGIN
            RESPONSE:='MOVED';
            SETCVAL(RESPONSE);
            SETCHAIN('EXEC/MOVE');
            GOTOXY(42,10);
            WRITELN('PLEASE WAIT, PROCESSING');
            POKE(0,-16299);
            EXIT(PROGRAM);
          END
        ELSE ADD:=TRUE;
      END;

    7:BEGIN
        RESPONSE:='';
        SETCVAL(RESPONSE);
        LEAVE:=TRUE;
```

```
      END;
  END;
END;

PROCEDURE INIT;
BEGIN
  GETCVAL(RESPONSE);
  IF RESPONSE='KRUNCHED' THEN
    BEGIN
      NEW:=TRUE;
      ADD:=FALSE;
      COUNT:=1;
    END
  ELSE
    BEGIN
      IF RESPONSE='MOVED' THEN
        BEGIN
          NEW:=FALSE;
          ADD:=TRUE;
          COUNT:=6;
        END
      ELSE
        BEGIN
          NEW:=FALSE;
          ADD:=FALSE;
          COUNT:=0;
        END
    END;
  LOOK:=FALSE;
  EVERY:=TRUE;
  LEAVE:=FALSE;
  LIST:=FALSE;
  SPECIFIC:=FALSE;
END;

PROCEDURE CHECK;
BEGIN
  IF COUNT=0 THEN
    BEGIN
      WIPE;
      WRITELN('POINT.ONE ALREADY EXISTS');
      WRITELN;
      WRITE('PRESS <CR> TO CONTINUE');
      READLN;
    END;
  IF IORESULT<>0 THEN
    BEGIN
      COUNT:=0;
      WIPE;
      WRITELN('ONE OF YOUR FILES DOES NOT EXIST');
      WRITELN;
      WRITELN('PRESS <CR> TO CONTINUE');
      READLN;
    END;
END;

PROCEDURE SETUP;
BEGIN
```

```
PROMPT[1]:='START VALUE ';
PROMPT[2]:='STOP  VALUE ';
REPEAT
  INIT;
  ERASE;
  WRITELN('STRUCTURE MAIN MENU - EACH ENTRY MUST ');
  WRITELN;
  WRITELN('BE FOLLOWED BY A CARRIAGE RETURN <CR>');
  GOTOXY(0,5);
  WRITELN('1. CREATE A NEW SET OF FILES');
  WRITELN;
  WRITELN('2. LOOK AT ALL OF THE DATA');
  WRITELN;
  WRITELN('3. LOOK AT SPECIFIC RECORDS');
  WRITELN;
  WRITELN('4. PRINT OUT ALL OF THE DATA');
  WRITELN;
  WRITELN('5. PRINT OUT SPECIFIC RECORDS');
  WRITELN;
  WRITELN('6. ADD TO THE FILE');
  WRITELN;
  WRITELN('7. EXIT THE PROGRAM');
  LINES(2);
  GETCVAL(RESPONSE);
  IF (RESPONSE='KRUNCHED') OR (RESPONSE='MOVED') THEN
    BEGIN
      RESPONSE:='RUN';
      SETCVAL(RESPONSE);
    END
  ELSE
    BEGIN
      REPEAT
        WRITE('ENTER THE NUMBER OF YOUR CHOICE ');
        READLN(COUNT);
        WIPE;
        IF NOT (COUNT IN [1..7]) THEN
          BEGIN
            WRITELN('PLEASE ENTER A NUMBER BETWEEN 1 AND 7');
            WRITELN;
            WRITE('PRESS <CR> TO CONTINUE');
            READLN;
            WIPE;
            WIPE;
            WIPE;
          END;
      UNTIL COUNT IN [1..7];
      FIND;
    END;
  IF LEAVE THEN EXIT(SETUP);
  WRITELN('FILES ARE BEING OPENED...PLEASE WAIT');
  IF NEW THEN
    BEGIN
      H^:=0;
      RESET(H,'#5:POINT.ONE');
      IF IORESULT<>0 THEN
        BEGIN
          REWRITE(H,'#5:POINT.ONE.COUNT');
          PUT(H);
```

```
              CLOSE(H,LOCK);
              RESET(H,'#5:POINT.ONE.COUNT');
              SEEK(H,0);
              REWRITE(F,'#5:POINT.ONE');
            END
          ELSE
            BEGIN
              CLOSE(H,LOCK);
              COUNT:=0;
            END;
        END
      ELSE
        BEGIN
          RESET(H,'#5:POINT.ONE.COUNT');
          IF IORESULT=0 THEN RESET(F,'#5:POINT.ONE');
        END;
      CHECK;
    UNTIL COUNT<>0;
  END;

  PROCEDURE RECDSET;
  BEGIN
    WIPE;
    IF LIST THEN REWRITE(OUT,'PRINTER:')
    ELSE REWRITE(OUT,'CONSOLE:');
    IF NOT NEW THEN
      BEGIN
        SEEK(H,0);
        GET(H);
        IF (LOOK) AND (NOT SPECIFIC)  THEN
          BEGIN
            STOP:=H^;
            H^:=0;
          END
        ELSE
          BEGIN
            IF SPECIFIC THEN
              BEGIN
                REPEAT
                  WRITE('ENTER START RECORD NUMBER ');
                  READLN(START);
                  WIPE;
                  IF NOT (START IN [0..(H^-1)]) THEN
                    BEGIN
                      WRITELN('THIS RECORD DOES NOT EXIST..RE-ENTER');
                      WRITELN;
                      WRITE('HIT <CR> TO CONTINUE');
                      READLN;
                      WIPE;
                      WIPE;
                      WIPE;
                    END;
                UNTIL START IN [0..(H^-1)];
                REPEAT
                  WRITE('ENTER STOP RECORD NUMBER ');
                  READLN(STOP);
                  WIPE;
                  IF NOT (STOP IN [START..(H^-1)]) THEN
```

```
                BEGIN
                  WRITE('PLEASE ENTER A NUMBER BETWEEN ',START);
                  WRITELN(' AND ',(H^-1));
                  WRITELN;
                  WRITE('HIT <CR> TO CONTINUE');
                  READLN;
                  WIPE;
                  WIPE;
                  WIPE;
                END;
              UNTIL STOP IN [START..(H^-1)];
              STOP:=STOP+1;
              H^:=START;
            END;
        END;
        SEEK(F,H^);
    END;

  IF (NOT NEW) AND (NOT ADD) THEN
    BEGIN
      WRITE('SHOULD I EXECUTE EVERY PROCEDURE? ');
      READLN(RESPONSE);
      IF RESPONSE<>'Y' THEN
        BEGIN
          ERASE;
          WRITELN('1. SEQUENCE ONE INFORMATION');
          WRITELN;
          WRITELN('2. TEXT OF CHOICES');
          WRITELN;
          WRITELN('3. POSITION OF CHOICES');
          WRITELN;
          WRITELN('4. CORRECT CHOICE');
          WRITELN;
          WRITELN('5. TEXT BEFORE/AFTER MOTION SEQUENCE');
          WRITELN;
          WRITELN('6. NUMBER OF PAGES OF TEXT');
          WRITELN;
          WRITELN('7. PICK UP A RESPONSE');
          WRITELN;
          WRITELN('8. INFORMATION PERTAINING TO ANSWERS');
          LINES(2);
          REPEAT
            WRITE('ENTER THE PROCEDURE NUMBER ');
            READLN(VALUE);
            WIPE;
            IF NOT (VALUE IN [1..8]) THEN
              BEGIN
                WRITELN('PLEASE ENTER A VALUE BETWEEN 1 AND 8');
                WRITELN;
                WRITE('PRESS <CR> TO CONTINUE');
                READLN;
                WIPE;
                WIPE;
                WIPE;
              END;
          UNTIL VALUE IN [1..8];
          EVERY:=FALSE;
        END;
```

```
    END;
END;

BEGIN
  SETUP;
  IF NOT LEAVE THEN RECDSET;
END;
```

Reset

```
PROGRAM RSET;

(* THIS PROGRAM RESETS THE COUNT FIELD IN POINT.ONE AND DELETES *)
(* ALL STUDENT FILES.                                           *)

USES LIGHTPEN,WORK;

(* F IS THE MAIN POINTER FILE *)
VAR F:FILE OF RECORD
        SEQ1:PACKED ARRAY[1..2]OF STRING[5];
        SEQ2:PACKED ARRAY[1..5, 1..2]OF STRING[5];
        SEQ3:PACKED ARRAY[1..5, 1..4]OF STRING[5];
        ANSWERS:PACKED ARRAY[1..20]OF STRING[40];
        POSITIONS:ARRAY[1..5,1..2]OF INTEGER;
        SEQMANY:PACKED ARRAY[1..5]OF INTEGER;
        FEEDMANY:PACKED ARRAY[1..5]OF INTEGER;
        FEEDWHERE:PACKED ARRAY[1..5]OF INTEGER;
        CORRECT:INTEGER;
        TEXTMANY:PACKED ARRAY[1..2]OF INTEGER;
        TEXTWHERE:INTEGER;
        RESPONSE:BOOLEAN;
        NEXT:PACKED ARRAY[1..5]OF INTEGER;
        DONE:PACKED ARRAY[1..5]OF BOOLEAN;
        TEXT:PACKED ARRAY[1..2,1..2]OF BOOLEAN;
        PLAY:PACKED ARRAY[1..2,1..2]OF BOOLEAN;
      END;

    G:FILE OF RECORD
        NUMBER:INTEGER;
        CHOSEN:PACKED ARRAY[1..10]OF INTEGER;
        TIMERS:PACKED ARRAY[1..10]OF INTEGER;
        KEEP:PACKED ARRAY[1..10]OF BOOLEAN;
      END;
    I, J:INTEGER;
    M:FILE OF PACKED ARRAY[1..1]OF STRING[14];
    NAME:STRING;
    ANSWER:CHAR;

BEGIN
  ERASE;
  GOTOXY(0,14);
  RESET(F,'#5:POINT.ONE');
  SEEK(F,0);
  GET(F);
  IF F^.TEXTMANY[2]>0 THEN
    BEGIN
      WRITELN('POINT.ONE HAS BEEN RESET');
      WRITELN;
      WRITELN('AND THE FOLLOWING FILES DELETED');
      LINES(2);
      RESET(M,'#5:STUDENT');
      SEEK(M,0);
      FOR J:=1 TO F^.TEXTMANY[2] DO
        BEGIN
          GET(M);
          NAME:=COPY(M^[1],4,11);
          WRITELN(NAME);
          RESET(G,M^[1]);
```

```
        CLOSE(G,PURGE);
      END;
    F^.TEXTMANY[2]:=0;
    SEEK(F,0);
    PUT(F);
    CLOSE(F,LOCK);
    CLOSE(M,LOCK);
  END
ELSE WRITELN('POINT.ONE IS ALREADY RESET');
END.
```

Look

```
(*$S+*)

PROGRAM LOOK;

USES LIGHTPEN,WORK;

VAR F:FILE OF RECORD
        SEQ1:PACKED ARRAY[1..2]OF STRING[5];
        SEQ2:PACKED ARRAY[1..5, 1..2]OF STRING[5];
        SEQ3:PACKED ARRAY[1..5, 1..4]OF STRING[5];
        ANSWERS:PACKED ARRAY[1..20]OF STRING[40];
        POSITIONS:ARRAY[1..5,1..2]OF INTEGER;
        SEQMANY:PACKED ARRAY[1..5]OF INTEGER;
        FEEDMANY:PACKED ARRAY[1..5]OF INTEGER;
        FEEDWHERE:PACKED ARRAY[1..5]OF INTEGER;
        CORRECT:INTEGER;
        TEXTMANY:PACKED ARRAY[1..2]OF INTEGER;
        TEXTWHERE:INTEGER;
        RESPONSE:BOOLEAN;
        NEXT:PACKED ARRAY[1..5]OF INTEGER;
        DONE:PACKED ARRAY[1..5]OF BOOLEAN;
        TEXT:PACKED ARRAY[1..2,1..2]OF BOOLEAN;
        PLAY:PACKED ARRAY[1..2,1..2]OF BOOLEAN;
      END;

    G:FILE OF RECORD
        NUMBER:INTEGER;
        CHOSEN:PACKED ARRAY[1..10]OF INTEGER;
        TIMERS:PACKED ARRAY[1..10]OF INTEGER;
        KEEP:PACKED ARRAY[1..10]OF BOOLEAN;
      END;

    M:FILE OF PACKED ARRAY[1..1]OF STRING[14];

    EXTENT, CHOICE, I, J, K, L:INTEGER;
    STATE:ARRAY[1..5] OF STRING;
    ID, NAME:STRING;
    OUT:INTERACTIVE;
    PEDAGO:BOOLEAN;
    QUIT, ANSWER:CHAR;

PROCEDURE START;
BEGIN
  STATE[1]:='GO BACK TO THE BEGINNING';
  STATE[2]:='GO BACK ONE JUNCTION POINT';
  STATE[3]:='REPEAT THE LAST JUNCTION POINT';
  STATE[4]:='QUIT THE PROGRAM';
  STATE[5]:='IGNORE THE HELP REQUEST';
  ERASE;
  WRITE('USE THE PRINTER? ');
  READ(ANSWER);
  IF ANSWER='Y' THEN REWRITE(OUT,'PRINTER:')
  ELSE REWRITE(OUT,'CONSOLE:');
  RESET(F,'#5:POINT.ONE');
  SEEK(F,0);
  GET(F);
  RESET(M,'#5:STUDENT');
  SEEK(M,0);
```

```
END;



PROCEDURE TIMEPRINT;
BEGIN
  ERASE;
  GET(M);
  ID:=M^[1];
  RESET(G,ID);
  SEEK(G,0);
  GET(G);
  PEDAGO:=G^.KEEP[1];
  ID:=COPY(ID,4,11);
  WRITELN(OUT,'SERIAL NUMBER: ',ID);
  IF PEDAGO THEN WRITELN(OUT,'PEDAGOGICAL MODE')
  ELSE WRITELN(OUT,'EXPERIENTIAL MODE');
  EXTENT:=G^.NUMBER;
  WITH G^ DO
    BEGIN
      WRITE(OUT,'START TIME: ',TIMERS[1],':',TIMERS[2]);
      WRITELN(OUT,':',TIMERS[3]);
      WRITE(OUT,'STOP TIME:  ',TIMERS[4],':',TIMERS[5]);
      WRITELN(OUT,':',TIMERS[6]);
      IF TIMERS[1]<>TIMERS[4] THEN TIMERS[5]:=TIMERS[5]+60;
      TIMERS[5]:=TIMERS[5]*60;
      TIMERS[5]:=TIMERS[5]+TIMERS[6];
      TIMERS[2]:=TIMERS[2]*60;
      TIMERS[2]:=TIMERS[2]+TIMERS[3];
      TIMERS[5]:=TIMERS[5]-TIMERS[2];
      WRITELN(OUT,'TOTAL TIME: ',TIMERS[5],' SECONDS');
    END;
  WRITELN(OUT);
  IF ANSWER<>'Y' THEN
    BEGIN
      READLN;
      ERASE;
    END;
END;

PROCEDURE ZERO;
BEGIN
  WITH G^ DO
    BEGIN
      IF NUMBER=0 THEN
        BEGIN
          IF K=1 THEN
            BEGIN
              WRITELN(OUT);
              IF PEDAGO THEN
                BEGIN
                  WRITELN(OUT,'HELP WAS SELECTED.');
                  WRITE(OUT,'THE OPTION SELECTED ');
                  WRITELN(OUT,'WHILE IN HELP WAS:');
                END
              ELSE
                BEGIN
                  WRITELN(OUT,'AN END POINT WAS REACHED');
```

```
                    WRITELN(OUT,'THE OPTION SELECTED WAS:');
                  END;
              END;
          END


        ELSE
          BEGIN
            IF K=1 THEN
              BEGIN
                WRITELN(OUT);
                WRITELN(OUT,'JUNCTION POINT ',NUMBER);
                IF PEDAGO THEN WRITELN(OUT,'PRIOR TO CORRECT CHOICE:');
              END
            ELSE
              BEGIN
                IF PEDAGO THEN
                  BEGIN
                    WRITELN(OUT,'FOLLOWING CORRECT CHOICE:');
                    IF CHOSEN[1]=0 THEN WRITELN(OUT,'NONE CHOSEN');
                  END;
              END;
          END;
      END;
  END;

  BEGIN
    START;
    FOR L:=1 TO F^.TEXTMANY[2] DO
      BEGIN
        TIMEPRINT;
        FOR I:=1 TO EXTENT DO
          BEGIN
            WITH G^ DO
              BEGIN
                FOR K:=1 TO 2 DO
                  BEGIN
                    GET(G);
                    ZERO;
                    J:=1;
                    WHILE (CHOSEN[J]<>0) AND (J<11) DO
                      BEGIN
                        IF NUMBER=0 THEN WRITELN(OUT,STATE[(CHOSEN[J])])
                        ELSE
                          BEGIN
                            WRITE(OUT,'CHOICE #',CHOSEN[J]);
                            IF KEEP[J] THEN WRITE(OUT,' KEPT    ')
                            ELSE WRITE(OUT,' NOT KEPT');
                            IF TIMERS[J]=1 THEN
                              WRITELN(OUT,' (',TIMERS[J],' SECOND)')
                            ELSE
                              WRITELN(OUT,' (',TIMERS[J],' SECONDS)');
                          END;
                        J:=J+1;
                      END;
                  END;
              END;
            IF ANSWER<>'Y' THEN
```

```
                BEGIN
                  READLN;
                  ERASE;
                END;
            END;
          CLOSE(G,LOCK);
          ERASE;
          WRITELN(OUT);
          WRITELN(OUT);
          WRITELN(OUT);
          WRITELN(OUT);
        END;
      CLOSE(F,LOCK);
      CLOSE(M,LOCK);
    END.
```

Align

```
PROGRAM ALIGN;

USES LIGHTPEN, WORK;

VAR EQUAL, SPACE, XCH, YCH, I, X, Y:INTEGER;

PROCEDURE STUFF;
BEGIN
  POKE(SPACE,1024);
  POKE(XCH,1025);
  POKE(EQUAL,1026);
  PLACE(X,1027);
  POKE(SPACE,1029);
  POKE(YCH,1030);
  POKE(EQUAL,1031);
  PLACE(Y,1032);
  POKE(SPACE,1034);
END;

BEGIN
  SPACE:=160;
  EQUAL:=253;
  XCH:=216;
  YCH:=217;
  ERASE;
  FOR I:=1 TO 23 DO
    BEGIN
      WRITELN('****************************************');
    END;
  GOTOXY(0,0);
  FOR I:=1 TO 2 DO
    BEGIN
      WRITELN(' ':40);
    END;
  FOR I:=5 TO 23 DO
    BEGIN
      GOTOXY(0,(I-1));
      WRITE(I);
    END;
  GOTOXY(0,2);
  FOR I:=1 TO 40 DO
    BEGIN
      IF I<10 THEN WRITE(I)
      ELSE
        BEGIN
          X:=I;
          X:=X DIV 10;
          WRITE(X);
        END;
    END;
  GOTOXY(0,3);
  WRITE(' ':9);
  GOTOXY(9,3);
  FOR X:=1 TO 4 DO
    BEGIN
      FOR I:=1 TO 10 DO
        BEGIN
          WRITE(I-1);
```

```
      END;
    END;
  GOTOXY(10,11);
  WRITE(' PRESS PEN TO LINE 20 TO EXIT  ');
  REPEAT
    PENXY(TRUE);
    Y:=YHIRES;
    Y:=Y DIV 8;
    Y:=Y+2;
    X:=XHIRES;
    X:=X DIV 7;
    X:=X+1;
    STUFF;
  UNTIL Y=20;
  ERASE;
END.
```

Unit Work

```
(*$S+*)
UNIT WORK; INTRINSIC CODE 27 DATA 28;

INTERFACE

USES LIGHTPEN;
PROCEDURE HUNT(VAR X,Y:INTEGER);
PROCEDURE ERASE;
PROCEDURE LINES(MANY:INTEGER);
FUNCTION RINGTOUCH:BOOLEAN;
PROCEDURE PEEK( VAR VALUE,ADDR:INTEGER);
PROCEDURE POKE(VALUE,ADDR:INTEGER);
PROCEDURE PLACE(CURRENT,ADDR:INTEGER);

IMPLEMENTATION

TYPE WINDOW=PACKED ARRAY[0..0] OF 0..255;
VAR HOLDX,HOLDY,HOLD,VALUE,I,ADDRESS:INTEGER;
    P:^WINDOW;

PROCEDURE HUNT;
BEGIN
  HOLDX:=0;
  HOLDY:=0;
  HOLD:=0;
  WHILE HOLD<>5 DO
    BEGIN
      PENXY(TRUE);
      Y:=YHIRES;
      Y:=Y DIV 8;
      Y:=Y+2;
      X:=XHIRES;
      X:=X DIV 7;
      X:=X+1;
      IF (Y=HOLDY) AND (X=HOLDX) THEN HOLD:=HOLD+1;
      HOLDX:=X;
      HOLDY:=Y;
    END;
END;

PROCEDURE ERASE;
BEGIN;
  WRITE(CHR(12));
END;

PROCEDURE LINES;
BEGIN
  FOR I:=1 TO MANY DO
    BEGIN
      WRITELN;
    END;
END;

FUNCTION RINGTOUCH;
BEGIN
  ADDRESS:=-16170;
  MOVELEFT(ADDRESS,P,2);
  IF P^[0]>128 THEN RINGTOUCH:=TRUE
  ELSE RINGTOUCH:=FALSE;
```

```
END;



PROCEDURE PEEK;
BEGIN
  MOVELEFT(ADDR,P,2);
  VALUE:=P^[0];
END;

PROCEDURE POKE;
BEGIN
  MOVELEFT(ADDR,P,2);
  P^[0]:=VALUE;
END;

PROCEDURE PLACE;
BEGIN
  IF CURRENT<10 THEN
    BEGIN
      POKE(160,ADDR);
      VALUE:=CURRENT+176;
      ADDR:=ADDR+1;
      POKE(VALUE,ADDR);
    END
  ELSE
    BEGIN
      HOLD:=CURRENT;
      CURRENT:=HOLD DIV 10;
      VALUE:=CURRENT+176;
      POKE(VALUE,ADDR);
      CURRENT:=HOLD-(CURRENT*10);
      VALUE:=CURRENT+176;
      ADDR:=ADDR+1;
      POKE(VALUE,ADDR);
    END;
END;


BEGIN

END.
```

APPENDIX L

EVALUATION INSTRUMENTS

James E. Schroeder, Ph.D.

U.S. Army Research Institute

Fort Benning Field Unit

APPENDIX L

EVALUATION INSTRUMENTS

Table of Contents

Part		Page
1.	Leadership Principles Test.	L-2
2.	IOBC Leadership Training Preference	L-5

TEST

Name_____ Group_____ S#_____

Please look at all three questions before you begin this test.

1. In the last 50 minutes, you have learned (or been reminded) of certain "principles" that you as a leader should remember when dealing with people. List as many of these "principles" as you can remember in the space below (use the back of the page if necessary). An example of a "principle" might be: "If you see a soldier doing an outstanding job, see that he gets some recognition". List only those principles which were taught in the last 50 min.

1.

2.

3.

4.

5.

6.

7.

8.

9.

10.

11.

12.

13.

(Use back of sheet if necessary.)

2. In the last 50 munutes, you have learned (or been reminded) of certain "principles" that should help you in performing your duties as an Officer. List as many of these "principles" as you can remember in the space below (use the back of the page if necessary). An example of a "principle" might be: "Always make a written record of important meetings".
List only those principles which were taught in the last 50 min.

1.

2.

3.

4.

5.

6.

7.

8.

9.

10.

11.

12.

13.

(Use back of sheet if necessary.)

3. In the last 50 minutes, you have learned (or been reminded) of certain "facts" about how the Army operates. List as many of these "facts" as you can remember in the space below (use the back if necessary). An example of such a "fact" might be: "The Platoon Leader is responsible for all issued property that is sub-hand receipted from the Company Commander to him". List only those principles which were taught in the last 50 min.

1.

2.

3.

4.

5.

6.

7.

8.

9.

10.

11.

12.

13.

(Use back of sheet if necessary.)

Appendix L

Part 2

Name _____

Service # _____

Group _____

S# _____

IOBC

LEADERSHIP TRAINING

PREFERENCE

1. How much do you think you would learn about dealing with people using the following approaches: (Circle one in each row)

	Nothing								A great deal
Role Playing:	1	2	3	4	5	6	7	8	9
Textbook:	1	2	3	4	5	6	7	8	9
Videodisc:	1	2	3	4	5	6	7	8	9

2. How useful are the following three types of leadership training to Second Lieutenants like yourself? (Circle one in each row)

	Useless								Extremely useful
Videodisc:	1	2	3	4	5	6	7	8	9
Textbook:	1	2	3	4	5	6	7	8	9
Role Playing:	1	2	3	4	5	6	7	8	9

3. How did the training keep your attention? (Circle one in each row)

	Very boring. I lost interest.								It was very interesting.
Textbook:	1	2	3	4	5	6	7	8	9
Videodisc:	1	2	3	4	5	6	7	8	9
Role Playing:	1	2	3	4	5	6	7	8	9

4. How motivated did the training methods keep you?
(Circle one in each row)

	It didn't motivate me.								It was very motivating.
Textbook:	1	2	3	4	5	6	7	8	9
Role Playing:	1	2	3	4	5	6	7	8	9
Videodisc:	1	2	3	4	5	6	7	8	9

5. In terms of learning about Army leadership, I think that _____ is the best training approach, _____ is second best, and _____ is third.
(Fill in blanks with Textbook, Role Playing, and Videodisc.)

6. In terms of keeping my interest, I think that _____ is the best training approach, _____ is second best and _____ is third.
(Fill in the blanks with Textbook, Role Playing, and Videodisc.)

7. Would you prefer to see a mixture of the three approaches in the Counseling Laboratory?

YES NO

8. If yes, what %?

_____ Textbook
\+ _____ Videodisc
\+ _____ Role Playing
100%

In the Role Playing session did you (check one): _____ Role Play
or
_____ just watch

9. When you were shown the videodisc approach, you were shown two teaching techniques. The Pedagogical approach provided text feedback about why answers were the best or not the best and the Experiential allowed events go on without too much interruption (simulated a real interaction).

Which of these two do you think is more valuable in learning leadership?

_____ Experiential
or
_____ Pedagogical

10. Do you think they should be combined? _____ Yes or _____ No

If "Yes" how should they be combined?

11. Evaluate the following for the videodisc:

	Distracted from Training								Contributed to Training
Quality of Writing	1	2	3	4	5	6	7	8	9
Quality of Filming	1	2	3	4	5	6	7	8	9
Quality of Acting	1	2	3	4	5	6	7	8	9
Quality of Feedback	1	2	3	4	5	6	7	8	9

12. To what extent did you agree with the content of the training for the three approaches?

	Agreed with none of the Training								Agreed with all of the Training
Role Playing	1	2	3	4	5	6	7	8	9
Textbook	1	2	3	4	5	6	7	8	9
Videodisc	1	2	3	4	5	6	7	8	9

13. Do you have any comments or suggestions for leadership training or the way this research was conducted?

APPENDIX M

STATISTICAL RESULTS OF THE EVALUATION

James E. Schroeder, Ph.D.

Sid Hall*

John C. Morey, Ph.D.

U.S. Army Research Institute

Fort Benning Field Unit

*Mr. Sid Hall is a doctoral candidate at Auburn University working with ARI through the Cooperative Education Program at Auburn University.

APPENDIX M

STATISTICAL RESULTS OF THE EVALUATION

Table of Contents

Part		Page
1.	Verbal Abuse Evaluation	M-2
2.	Taking Charge: Meeting the Platoon Sergeant EValuation . .	M-6
3.	Taking Charge: Meeting the NCO's and the Platoon Evaluation	M-10
4.	Performance Counseling Evaluation	M-14
5.	Insubordination: Moderate and Severe Evaluation.	M-18
6.	Personal Crises: Emergency Leave and Suicide Threat Evaluation.	M-22
7.	Overall Evaluation Results.	M-26

Table 30

RESULTS OF EVALUATION 1
FOR
LEADERSHIP PRINCIPLES TEST

CONDESCRIPTIVE ON VISTA TEST DATA

SUBFILE VI

- O N E W A Y -

VARIABLE SCORE

ANALYSIS OF VARIANCE

SOURCE	D.F.	SUM OF SQUARES	MEAN SQUARES	F RATIO	F PROB.
BETWEEN GROUPS	2	52.4071	26.2035	2.244	0.1121
WITHIN GROUPS	87	1015.9490	11.6776		
TOTAL	89	1068.3560			

GROUP	COUNT	MEAN	STANDARD DEVIATION	STANDARD ERROR	MINIMUM	MAXIMUM	95 PCT CONF INT FOR MEAN
GRPC1(Videodisc)	30	11.5776	3.3159	0.6054	4.3300	17.3300	10.1395 TO 12.0158
GRPC2(Text)	29	10.3358	3.5440	0.6210	5.3300	19.6700	9.0611 TO 11.6051
GRPC3(Role Play)	31	8.5274	3.5776	0.6425	2.3300	17.0000	6.2151 TO 10.8397
TOTAL	90	10.4039	3.4647	0.3652	2.3300	19.6700	9.6781 TO 11.1294

TESTS FOR HOMOGENEITY OF VARIANCES

COCHRAN C = MAX. VARIANCE/SUM(VARIANCES) = 0.3659, P = 0.944 (APPROX.)
[illegible] = 0.[illegible], P = [illegible]
MAXIMUM VARIANCE / MINIMUM VARIANCE = 1.156

Agreement: Judge 1 and Judge 2 = 87%
Judge 1 and Judge 3 = 77%
Judge 2 and Judge 3 = 71%

(Judge 1 and Judge 2 were from the VISTA project, Judge 3 was not familiar with the project.)

Table 31

Evaluation 1 (Mean Ratings)

Note: * denotes $p < .05$
n.s. denotes $p > .05$

Name _____

Service # _____

Group _____

S# _____

IOBC

LEADERSHIP TRAINING

PREFERENCE

1. How much do you think you would learn about dealing with people using the following approaches: (Circle one in each row)

	Nothing								A great deal			
Role Playing:	1	2	3	4	5	6	7 ▲	8	9	6.75	*	
Textbook:	1	2	3	4	5 ▲	6	7	8	9	4.98		n.s. *
Videodisc:	1	2	3	4	5	6	7 ▲	8	9	6.61		

2. How useful are the following three types of leadership training to Second Lieutenants like yourself? (Circle one in each row)

	Useless								Extremely useful			
Videodisc:	1	2	3	4	5	6	7 ▲	8	9	6.89	*	
Textbook:	1	2	3	4	5 ▲	6	7	8	9	4.99		n.s. *
Role Playing:	1	2	3	4	5	6	7 ▲	8	9	7.04		

3. How did the training keep your attention? (Circle one in each row)

	Very boring. I lost interest.								It was very interesting.			
Textbook:	1	2	3	4	5 ▲	6	7	8	9	4.88	*	
Videodisc:	1	2	3	4	5	6	7 ▲	8	9	7.22		* n.s.
Role Playing:	1	2	3	4	5	6	7 ▲	8	9	7.03		

Evaluation 1 (Continued)

4. How motivated did the training methods keep you?
(Circle one in each row)

	It didn't motivate me.									It was very motivating.	
Textbook:	1	2	3	4	▲ 5	6	7	8	9	4.48	*
Role Playing:	1	2	3	4	5	6	▲ 7	8	9	6.52	*
Videodisc:	1	2	3	4	5	6	▲ 7	8	9	7.05	n.s.

5. In terms of learning about Army leadership, I think that Role Play (42) is the best training approach, Videodisc (36) is second best, and Text (6) is third.
(Fill in blanks with Textbook, Role Playing, and Videodisc.)

6. In terms of keeping my interest, I think that Videodisc (47) is the best training approach, Role Play (33) is second best and Text (3) is third.
(Fill in the blanks with Textbook, Role Playing, and Videodisc.)

7. Would you prefer to see a mixture of the three approaches in the Counseling Laboratory?

YES	NO
92.1%	7.1%

8. If yes, what %?

	19.0%	Textbook
+	39.0%	Videodisc
+	41.9%	Role Playing
	100%	

In the Role Playing session did you (check one): 30.4% Role Play or 69.6% just watch

9. When you were shown the videodisc approach, you were shown two teaching techniques. The Pedagogical approach provided text feedback about why answers were the best or not the best and the Experiential allowed events go on without too much interruption (simulated a real interaction).

Which of these two do you think is more valuable in learning leadership?

37.0% Experiential
or
63.0% Pedagogical

(Note: only Videodisc subjects)

10. Do you think they should be combined? 70.4% Yes or 29.6% No

If "Yes" how should they be combined?

11. Evaluate the following for the videodisc:

	Distracted from Training								Contributed to Training (Medians)
Quality of Writing	1	2	3	4	5	6	▲ 7	8	9 (6.81)
Quality of Filming	1	2	3	4	5	6	7 ▲	8	9 (7.33)
Quality of Acting	1	2	3	4	5	6	▲ 7	8	9 (6.71)
Quality of Feedback	1	2	3	4	5	6	7 ▲	8	9 (7.40)

12. To what extent did you agree with the content of the training for the three approaches?

	Agreed with none of the Training								Agreed with all of the Training
Role Playing	1	2	3	4	5	6	▲ 7	8	9 (6.86)
Textbook	1	2	3	4	5	▲ 6	7	8	9 (5.79)
Videodisc	1	2	3	4	5	6	7 ▲	8	9 (7.04)

* n.s.

13. Do you have any comments or suggestions for leadership training or the way this research was conducted?

Table 32

RESULTS OF EVALUATION 2
FOR
LEADERSHIP PRINCIPLES TEST

CONDESCRIPTIVE ON VISTA TEST DATA

SUBFILE V2

- O N E W A Y -

VARIABLE SCORE

ANALYSIS OF VARIANCE

SOURCE	D.F.	SUM OF SQUARES	MEAN SQUARES	F RATIO	F PROB.
BETWEEN GROUPS	2	91.9199	45.9600	3.091	0.0540
WITHIN GROUPS	51	758.2278	14.8672		
TOTAL	53	850.1477			

GROUP	COUNT	MEAN	STANDARD DEVIATION	STANDARD ERROR	MINIMUM	MAXIMUM	95 PCT CONF INT FOR MEAN
GRPC1 (Role Play)	17	11.5117	4.3297	1.0501	5.3300	20.3300	9.2838 TO 13.7395
GRPC2 (Test)	16	8.7430	4.2526	1.0632	1.0000	15.0000	6.4753 TO 10.9547
GRPC3 (Role Play)	17	9.0535	2.6621	0.6457	4.0000	13.6700	7.6901 TO 10.4276
TOTAL	54	9.8515	4.0051	0.5450	1.0000	20.3300	8.7583 TO 10.9446

TESTS FOR HOMOGENEITY OF VARIANCES

COCHRAN'S C = MAX. VARIANCE/SUM(VARIANCES) = 0.4269, P = 0.673 (APPROX.)
BARTLETT-BOX F = 2.122, P = 0.129
MAXIMUM VARIANCE / MINIMUM VARIANCE = 2.664

Agreement: Judge 1 and Judge 2 = 83%
Judge 1 and Judge 3 = 66%
Judge 2 and Judge 3 = 58%

(Judge 1 and Judge 2 were from the VISTA project. Judge 3 was not familiar with the project.)

Table 33

Evaluation 2 (Mean Ratings)

Name _____

Service # _____

Group _____

S# _____

Note: * denotes $p < .05$
n.s. denotes $p > .05$

IOBC

LEADERSHIP TRAINING

PREFERENCE

1. How much do you think you would learn about dealing with people using the following approaches: (Circle one in each row)

	Nothing								A great deal			
Role Playing:	1	2	3	4	5	6	7▲	8	9	7.02	*	
Textbook:	1	2	3	4	5▲	6	7	8	9	4.56		n.s.
Videodisc:	1	2	3	4	5	6	7▲	8	9	6.69		*

2. How useful are the following three types of leadership training to Second Lieutenants like yourself? (Circle one in each row)

	Useless								Extremely useful			
Videodisc:	1	2	3	4	5	6	7▲	8	9	6.94	*	
Textbook:	1	2	3	4	5▲	6	7	8	9	4.94		n.s.
Role Playing:	1	2	3	4	5	6	7▲	8	9	7.18		*

3. How did the training keep your attention? (Circle one in each row)

	Very boring. I lost interest.								It was very interesting.			
Textbook:	1	2	3	4▲	5	6	7	8	9	4.40	*	
Videodisc:	1	2	3	4	5	6	7▲	8	9	7.28		*
Role Playing:	1	2	3	4	5	6	7▲	8	9	7.24		n.s.

Evaluation 2 (Continued)

4. How motivated did the training methods keep you?
(Circle one in each row)

	It didn't motivate me.									It was very motivating.	
Textbook:	1	2	3	▲4	5	6	7	8	9	3.72	*
Role Playing:	1	2	3	4	5	6	▲7	8	9	6.96	*
Videodisc:	1	2	3	4	5	6	▲7	8	9	6.80	n.s.

5. In terms of learning about Army leadership, I think that Role Play (32) is the best training approach, Videodisc (14) is second best, and Text (2) is third.
(Fill in blanks with Textbook, Role Playing, and Videodisc.)

6. In terms of keeping my interest, I think that Role Play (28) is the best training approach, Videodisc (19) is second best and Text (1) is third.
(Fill in the blanks with Textbook, Role Playing, and Videodisc.)

7. Would you prefer to see a mixture of the three approaches in the Counseling Laboratory?

YES	NO
88.0%	12.0%

8. If yes, what %?

19.4%	Textbook
+ 36.4%	Videodisc
+ 44.1%	Role Playing
100%	

Evaluation 2 (Continued)

In the Role Playing session did you (check one): 58.3% Role Play or 41.7% just watch

9. When you were shown the videodisc approach, you were shown two teaching techniques. The Pedagogical approach provided text feedback about why answers were the best or not the best and the Experiential allowed events go on without too much interruption (simulated a real interaction).

Which of these two do you think is more valuable in learning leadership?

23.5% Experiential or 76.5% Pedagogical

(Note: only Videodisc subjects)

10. Do you think they should be combined? 88.2% Yes or 11.8% No

If "Yes" how should they be combined?

11. Evaluate the following for the videodisc:

	Distracted from Training								Contributed to Training (Median)
Quality of Writing	1	2	3	4	5	6	▲7	8	9(6.85)
Quality of Filming	1	2	3	4	5	6	7▲	8	9(7.40)
Quality of Acting	1	2	3	4	5	6	▲7	8	9(6.89)
Quality of Feedback	1	2	3	4	5	6	7▲	8	9(7.38)

12. To what extent did you agree with the content of the training for the three approaches?

	Agreed with none of the Training								Agreed with all of the Training	
Role Playing	1	2	3	4	5	6	7▲	8	9(7.23)	*
Textbook	1	2	3	4	5	▲6	7	8	n.s 9(5.98)	
Videodisc	1	2	3	4	5	6	7▲	8	* 9(7.26)	

13. Do you have any comments or suggestions for leadership training or the way this research was conducted?

Table 34

RESULTS OF EVALUATION 3
FOR
LEADERSHIP PRINCIPLES TEST

CONDESCRIPTIVE ON VISTA TEST DATA

SUBFILE V2

- O N E W A Y -

VARIABLE SCORE

ANALYSIS OF VARIANCE

SOURCE	D.F.	SUM OF SQUARES	MEAN SQUARES	F RATIO	F PROB.
BETWEEN GROUPS	2	16.3359	8.1685	0.511	0.6044
WITHIN GROUPS	35	559.6826	15.9909		
TOTAL	37	576.0193			

GROUP	COUNT	MEAN	STANDARD DEVIATION	STANDARD ERROR	MINIMUM	MAXIMUM	95 PCT CONF INT FOR MEAN
GRPC1(Videotape)	11	12.4136	3.3565	1.0119	5.3030	15.6700	5.3290 TO 12.7019
GRPC2(Text)	13	10.3638	3.6091	0.9611	5.6730	16.6700	8.3055 TO 12.4190
GRPC3(Role Play)	14	11.7950	4.2221	1.1285	3.3300	25.3330	9.3431 TO 14.6269
TOTAL	38	10.9733	3.9656	0.6401	3.3300	25.3330	9.6334 TO 12.2272

TESTS FOR HOMOGENEITY OF VARIANCES

COCHRANS C = MAX. VARIANCE/SUM(VARIANCES) = 0.5225, P = 0.169 (APPROX.)
BARTLETT-BOX F = 1.254, P = 0.296
MAX/MIN = VARIANCE / MINIMUM VARIANCE = 2.247

Agreement: Judge 1 and Judge 2 = 84%
Judge 1 and Judge 3 = 90%
Judge 2 and Judge 3 = 88%

(Judge 1 and Judge 2 were from the VISTA project, Judge 3 was not familiar with the project.)

Table 35

Evaluation 3 (Mean Ratings)

Note: * denotes $p < .05$
n.s. denotes $p > .05$

Name _____

Service # _____

Group _____

S# _____

IOBC

LEADERSHIP TRAINING

PREFERENCE

1. How much do you think you would learn about dealing with people using the following approaches: (Circle one in each row)

	Nothing								A great deal		
Role Playing:	1	2	3	4	5	6	7▲	8	9	7.32	*
Textbook:	1	2	3	4	5▲	6	7	8	9	5.00	n.s.
Videodisc:	1	2	3	4	5	6	7▲	8	9	6.76	*

2. How useful are the following three types of leadership training to Second Lieutenants like yourself? (Circle one in each row)

	Useless								Extremely useful		
Videodisc:	1	2	3	4	5	6	7▲	8	9	6.74	*
Textbook:	1	2	3	4	5▲	6	7	8	9	5.16	n.s.
Role Playing:	1	2	3	4	5	6	7	8▲	9	7.74	*

3. How did the training keep your attention? (Circle one in each row)

	Very boring. I lost interest.								It was very interesting.		
Textbook:	1	2	3	4▲	5	6	7	8	9	4.34	*
Videodisc:	1	2	3	4	5	6	7	8▲	9	7.45	*
Role Playing:	1	2	3	4	5	6	7	8▲	9	7.92	n.s.

Evaluation 3 (Continued)

4. How motivated did the training methods keep you?
(Circle one in each row)

	It didn't motivate me.							It was very motivating.		
Textbook:	1	2	3	4▲	5	6	7	8	9	3.97
Role Playing:	1	2	3	4	5	6	7▲	8	9	7.58
Videodisc:	1	2	3	4	5	6	7▲	8	9	7.03

* * n.s.

5. In terms of learning about Army leadership, I think that Role Play (25) is the best training approach, Videodisc (13) is second best, and Text (0) is third.
(Fill in blanks with Textbook, Role Playing, and Videodisc.)

6. In terms of keeping my interest, I think that Role Play (24) is the best training approach, Videodisc (14) is second best and Text (0) is third.
(Fill in the blanks with Textbook, Role Playing, and Videodisc.)

7. Would you prefer to see a mixture of the three approaches in the Counseling Laboratory?

YES	NO
89.5%	10.5%

8. If yes, what %?

17.1%	Textbook
35.3%	Videodisc
47.6%	Role Playing
100%	

In the Role Playing session did you (check one): 34.3% Role Play or 65.7% just watch

9. When you were shown the videodisc approach, you were shown two teaching techniques. The Pedagogical approach provided text feedback about why answers were the best or not the best and the Experiential allowed events go on without too much interruption (simulated a real interaction).

Which of these two do you think is more valuable in learning leadership?

9.1% Experiential or 90.9% Pedagogical

(Note: only Videodisc subjects)

10. Do you think they should be combined? 72.7% Yes or 27.3% No

If "Yes" how should they be combined?

11. Evaluate the following for the videodisc:

	Distracted from Training								Contributed to Training (Medians)
Quality of Writing	1	2	3	4	5	6	▲7	8	9 (6.80)
Quality of Filming	1	2	3	4	5	6	7	▲8	9 (7.63)
Quality of Acting	1	2	3	4	5	6	▲7	8	9 (6.80)
Quality of Feedback	1	2	3	4	5	6	7	▲8	9 (7.62)

12. To what extent did you agree with the content of the training for the three approaches?

	Agreed with none of the Training								Agreed with all of the Training
Role Playing	1	2	3	4	5	6	7	▲8	9 (7.92) *
Textbook	1	2	3	4	5	6	▲7	8	n.s. 9 (6.32) *
Videodisc	1	2	3	4	5	6	7	▲8	9 (7.70)

13. Do you have any comments or suggestions for leadership training or the way this research was conducted?

Table 36

RESULTS OF EVALUATION 4
FOR
LEADERSHIP PRINCIPLES TEST

CONDESCRIPTIVE ON VISTA TEST DATA

SUBFILE V6

- O N E W A Y -

VARIABLE SCORE

ANALYSIS OF VARIANCE

SOURCE	D.F.	SUM OF SQUARES	MEAN SQUARES	F RATIO	F PROB.
BETWEEN GROUPS	2	196.2131	98.1065	3.028	0.0595
WITHIN GROUPS	63	1328.2014	32.3951		
TOTAL	65	1524.4145			

GROUP	COUNT	MEAN	STANDARD DEVIATION	STANDARD ERROR	MINIMUM	MAXIMUM	95 PCT CONF INT FOR MEAN
GRPC1 (Vignette)	15	16.7133	5.1134	1.3202	4.0000	33.0000	13.8271 TO 19.5995
GRPC2 (Text)	15	11.7747	5.5045	1.3698	2.0000	23.0000	8.8412 TO 14.7161
GRPC3 (Role Play)	36	15.5633	5.6244	1.5032	9.3300	25.0000	13.3961 TO 17.8910
TOTAL	66	15.6675	5.9541	0.9976	2.0000	33.0000	12.9573 TO 16.4777

TESTS FOR HOMOGENEITY OF VARIANCES

COCHRAN'S C = MAX. VARIANCE/SUM(VARIANCES) = 0.3847, P = 0.390 (APPROX.)
BARTLETT-BOX F = 0.138, P = 0.871
MAXIMUM VARIANCE / MINIMUM VARIANCE = 1.329

Agreement: Judge 1 and Judge 2 = 96%
Judge 1 and Judge 3 = 94%
Judge 2 and Judge 3 = 95%

(Judge 1 and Judge 2 were from the VISTA project, Judge 3 was not familiar with the project.)

Table 37

Evaluation 4 (Mean Ratings)

Name _____

Service # _____

Note: * denotes p .05
n.s. denotes p .05

Group _____

S# _____

IOBC

LEADERSHIP TRAINING

PREFERENCE

1. How much do you think you would learn about dealing with people using the following approaches: (Circle one in each row)

	Nothing								A great deal		
Role Playing:	1	2	3	4	5	6	▲7	8	9	7.25	*
Textbook:	1	2	3	4	▲5	6	7	8	9	4.54	n.s. *
Videodisc:	1	2	3	4	5	▲6	7	8	9	6.25	

2. How useful are the following three types of leadership training to Second Lieutenants like yourself? (Circle one in each row)

	Useless								Extremely useful		
Videodisc:	1	2	3	4	5	6	▲7	8	9	6.52	*
Textbook:	1	2	3	4	▲5	6	7	8	9	4.93	* *
Role Playing:	1	2	3	4	5	6	7	▲8	9	7.77	

3. How did the training keep your attention? (Circle one in each row)

	Very boring. I lost interest.								It was very interesting.		
Textbook:	1	2	3	▲4	5	6	7	8	9	4.27	*
Videodisc:	1	2	3	4	5	6	▲7	8	9	6.91	* n.s.
Role Playing:	1	2	3	4	5	6	7	▲8	9	7.41	

Evaluation 4 (Continued)

4. How motivated did the training methods keep you?
(Circle one in each row)

	It didn't motivate me.									It was very motivating.			
Textbook:	1	2	3	▲4	5	6	7	8	9	3.66	*		
Role Playing:	1	2	3	4	5	6	7▲	8	9	7.14		*	n.s.
Videodisc:	1	2	3	4	5	6▲	7	8	9	6.41			

5. In terms of learning about Army leadership, I think that Role Play (23) is the best training approach, Videodisc (16) is second best, and Text (5) is third.
(Fill in blanks with Textbook, Role Playing, and Videodisc.)

6. In terms of keeping my interest, I think that Role Play (26) is the best training approach, Videodisc (17) is second best and Text (1) is third.
(Fill in the blanks with Textbook, Role Playing, and Videodisc.)

7. Would you prefer to see a mixture of the three approaches in the Counseling Laboratory?

YES	NO
90.5%	9.5%

8. If yes, what %?

	17.7%	Textbook
+	36.1%	Videodisc
+	45.9%	Role Playing
	100%	

In the Role Playing session did you (check one): 18.4% Role Play or 81.6% just watch

9. When you were shown the videodisc approach, you were shown two teaching techniques. The Pedagogical approach provided text feedback about why answers were the best or not the best and the Experiential allowed events go on without too much interruption (simulated a real interaction).

Which of these two do you think is more valuable in learning leadership?

28.6% Experiential or 71.4% Pedagogical

(Note: only Videodisc subjects)

10. Do you think they should be combined? 50.0% Yes or 50.0% No

If "Yes" how should they be combined?

11. Evaluate the following for the videodisc:

	Distracted from Training								Contributed to Training
Quality of Writing	1	2	3	4	5	6	▲ 7	8	9 (6.69)
Quality of Filming	1	2	3	4	5	6	▲ 7	8	9 (6.96)
Quality of Acting	1	2	3	4	▲ 5	6	7	8	9 (5.11)
Quality of Feedback	1	2	3	4	5	6	▲ 7	8	9 (7.31)

12. To what extent did you agree with the content of the training for the three approaches?

	Agreed with none of the Training								Agreed with all of the Training
Role Playing	1	2	3	4	5	6	▲ 7	8	9 (7.24) *
Textbook	1	2	3	4	▲ 5	6	7	8	n.s. 9 (5.64) *
Videodisc	1	2	3	4	5	6	▲ 7	8	9 (6.95)

13. Do you have any comments or suggestions for leadership training or the way this research was conducted?

Table 38

RESULTS OF EVALUATION 5
FOR
LEADERSHIP PRINCIPLES TEST

CONDESCRIPTIVE ON VISTA TEST DATA

SUBFILE VS

- O N E - W A Y -

VARIABLE SCORE

ANALYSIS OF VARIANCE

SOURCE	D.F.	SUM OF SQUARES	MEAN SQUARES	F RATIO	F PROB.
BETWEEN GROUPS	2	181.3352	90.6676	3.847	0.0292
WITHIN GROUPS	42	989.6593	23.5633		
TOTAL	44	1170.9944			

GROUP	COUNT	MEAN	STANDARD DEVIATION	STANDARD ERROR	MINIMUM	MAXIMUM	95 PCT CONF INT FOR MEAN
GRPC1 (Film)	15	14.9933	6.1626	1.5907	7.6700	30.6600	11.4905 TO 18.5560
GRPC2 (Text)	15	13.6900	4.2451	1.0961	6.6700	19.3300	8.5391 TO 15.2409
GRPC3 (Role Play)	16	13.4630	4.3736	1.0934	1.3300	15.6700	8.3322 TO 12.6277
TOTAL	45	12.3226	5.1538	0.7690	1.3300	30.6600	10.4726 TO 13.5723

TESTS FOR HOMOGENEITY OF VARIANCES

COCHRANS C = MAX. VARIANCE/SUM(VARIANCES) = 0.5257, P = 0.107 (APPROX.)
BARTLETT-BOX F = 1.516, P = 0.220
MAXIMUM VARIANCE / MINIMUM VARIANCE = 2.358

Agreement: Judge 1 and Judge 2 = 87%
Judge 1 and Judge 3 = 90%
Judge 2 and Judge 3 = 86%

(Judge 1 and Judge 2 were from the VISTA project, Judge 3 was not familiar with the project.)

Table 39

Evaluation 5 (Mean Ratings)

Name _____

Service # _____

Group _____

S# _____

Note: * denotes $p < .05$
n.s. denotes $p > .05$

IOBC

LEADERSHIP TRAINING

PREFERENCE

1. How much do you think you would learn about dealing with people using the following approaches: (Circle one in each row)

	Nothing 1	2	3	4	5	6	7	8	9 A great deal			
Role Playing:							▲			7.32	*	
Textbook:					▲					4.80	n.s.	*
Videodisc:							▲			6.79		

2. How useful are the following three types of leadership training to Second Lieutenants like yourself? (Circle one in each row)

	Useless 1	2	3	4	5	6	7	8	9 Extremely useful			
Videodisc:							▲			7.02	*	
Textbook:					▲					5.14		*
Role Playing:								▲		7.71		*

3. How did the training keep your attention? (Circle one in each row)

	Very boring. I lost interest. 1	2	3	4	5	6	7	8	9 It was very interesting.			
Textbook:					▲					4.98	*	
Videodisc:								▲		7.74		*
Role Playing:								▲		7.86		n.s.

Evaluation 5 (Continued)

4. How motivated did the training methods keep you?
(Circle one in each row)

	It didn't motivate me.								It was very motivating.		
Textbook:	1	2	3	4	▲ 5	6	7	8	9	4.57	*
Role Playing:	1	2	3	4	5	6	7 ▲	8	9	7.40	*
Videodisc:	1	2	3	4	5	6	7 ▲	8	9	7.17	n.s.

5. In terms of learning about Army leadership, I think that Role Play (22) is the best training approach, Videodisc (17) is second best, and Text (2) is third.
(Fill in blanks with Textbook, Role Playing, and Videodisc.)

6. In terms of keeping my interest, I think that Role Play (21) is the best training approach, Videodisc (20) is second best and Text (0) is third.
(Fill in the blanks with Textbook, Role Playing, and Videodisc.)

7. Would you prefer to see a mixture of the three approaches in the Counseling Laboratory?

YES	NO
95.0%	5.0%

8. If yes, what %?

	16.4%	Textbook
+	36.1%	Videodisc
+	47.4%	Role Playing
	100%	

Evaluation 5 (Continued)

In the Role Playing session did you (check one): 20.0% Role Play or 80.0% just watch

9. When you were shown the videodisc approach, you were shown two teaching techniques. The Pedagogical approach provided text feedback about why answers were the best or not the best and the Experiential allowed events go on without too much interruption (simulated a real interaction).

Which of these two do you think is more valuable in learning leadership?

23.1% Experiential
or
76.9% Pedagogical

(Note: only Videodisc subjects)

10. Do you think they should be combined? 38.5% Yes or 61.5% No

If "Yes" how should they be combined?

11. Evaluate the following for the videodisc:

	Distracted from Training								Contributed to Training (Medians)
Quality of Writing	1	2	3	4	5	6	7 ▲	8	9 (7.36)
Quality of Filming	1	2	3	4	5	6	7 ▲	8	9 (7.70)
Quality of Acting	1	2	3	4	5	6	7 ▲	8	9 (7.36)
Quality of Feedback	1	2	3	4	5	6	7 ▲	8	9 (7.34)

12. To what extent did you agree with the content of the training for the three approaches?

	Agreed with none of the Training								Agreed with all of the Training	
Role Playing	1	2	3	4	5	6	7 ▲	8	9 (7.50)	*
Textbook	1	2	3	4	5	6 ▲	7	8	n.s. 8 (6.38)	*
Videodisc	1	2	3	4	5	6	7 ▲	8	9 (7.52)	

13. Do you have any comments or suggestions for leadership training or the way this research was conducted?

Table 40

RESULTS OF EVALUATION 6
FOR
LEADERSHIP PRINCIPLES TEST

CONDESCRIPTIVE ON VISTA TEST DATA

SUBFILE V6

- O N E W A Y -

VARIABLE SCORE

ANALYSIS OF VARIANCE

SOURCE	D.F.	SUM OF SQUARES	MEAN SQUARES	F RATIO	F PROB.
BETWEEN GROUPS	2	14.0652	7.0326	0.536	0.6066
WITHIN GROUPS	38	527.6323	13.8851		
TOTAL	40	541.6975			

GROUP	COUNT	MEAN	STANDARD DEVIATION	STANDARD ERROR	MINIMUM	MAXIMUM	95 PCT CONF INT FOR MEAN
GRP01 (Discussion)	14	10.7857	3.8635	0.8231	5.6700	15.6700	9.0147 TO 12.5581
GRP02 (Test)	14	11.5027	4.6452	1.2415	3.0000	18.6700	8.8186 TO 14.1828
GRP03 (Role Play)	13	12.2308	3.2232	0.8941	5.6700	16.6700	10.2826 TO 14.1789
TOTAL	41	11.6683	3.6900	0.5747	3.0000	18.6700	10.3267 TO 12.6498

TESTS FOR HOMOGENEITY OF VARIANCES

COCHRANS C = MAX. VARIANCE/SUM(VARIANCES) = 0.5214, P = 0.132 (APPROX.)
BARTLETT-BOX F = 1.350, P = 0.259
MAXIMUM VARIANCE / MINIMUM VARIANCE = 2.292

Agreement: Judge 1 and Judge 2 = 93%
Judge 1 and Judge 3 = 90%
Judge 2 and Judge 3 = 91%

(Judge 1 and Judge 2 were from the VISTA project, Judge 3 was not familiar with the project.)

Table 41

Evaluation 6 (Mean Ratings)

Name _____

Service # _____

Group _____

S# _____

Note: * denotes $p < .05$
n.s. denotes $p > .05$

IOBC

LEADERSHIP TRAINING

PREFERENCE

1. How much do you think you would learn about dealing with people using the following approaches: (Circle one in each row)

	Nothing								A great deal		
Role Playing:	1	2	3	4	5	6	7▲	8	9	7.17	*
Textbook:	1	2	3	4▲	5	6	7	8	9	4.05	n.s. *
Videodisc:	1	2	3	4	5	6	7▲	8	9	6.51	

2. How useful are the following three types of leadership training to Second Lieutenants like yourself? (Circle one in each row)

	Useless								Extremely useful		
Videodisc:	1	2	3	4	5	6▲	7	8	9	6.34	*
Textbook:	1	2	3	4▲	5	6	7	8	9	4.20	n.s. *
Role Playing:	1	2	3	4	5	6	7▲	8	9	7.19	

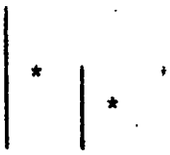
3. How did the training keep your attention? (Circle one in each row)

	Very boring. I lost interest.								It was very interesting.		
Textbook:	1	2	3▲	4	5	6	7	8	9	3.29	*
Videodisc:	1	2	3	4	5	6	7▲	8	9	6.90	*
Role Playing:	1	2	3	4	5	6	7	8▲	9	7.45	n.s.

Evaluation 6 (Continued)

4. How motivated did the training methods keep you?
(Circle one in each row)

	It didn't motivate me.								It was very motivating.	
Textbook:	1	2	▲3	4	5	6	7	8	9	2.73
Role Playing:	1	2	3	4	5	6	7▲	8	9	7.26
Videodisc:	1	2	3	4	5	6▲	7	8	9	6.15



5. In terms of learning about Army leadership, I think that Role Play (28) is the best training approach, Videodisc (13) is second best, and Text (0) is third.
(Fill in blanks with Textbook, Role Playing, and Videodisc.)

6. In terms of keeping my interest, I think that Role Play (27) is the best training approach, Videodisc (15) is second best and Text (0) is third.
(Fill in the blanks with Textbook, Role Playing, and Videodisc.)

7. Would you prefer to see a mixture of the three approaches in the Counseling Laboratory?

YES	NO
85.7%	14.3%

8. If yes, what %?

	13.8%	Textbook
+	35.4%	Videodisc
+	50.7%	Role Playing
	100%	

Evaluation 6 (Continued)

In the Role Playing session did you (check one): 27.3% Role Play or 72.7% just watch

9. When you were shown the videodisc approach, you were shown two teaching techniques. The Pedagogical approach provided text feedback about why answers were the best or not the best and the Experiential allowed events go on without too much interruption (simulated a real interaction).

Which of these two do you think is more valuable in learning leadership?

41.7% Experiential or 58.3% Pedagogical

(Note: only Videodisc subjects)

10. Do you think they should be combined? 76.9% Yes or 23.1% No

If "Yes" how should they be combined?

11. Evaluate the following for the videodisc:

	Distracted from Training								Contributed to Training (Medians)
Quality of Writing	1	2	3	4	5	6	▲7	8	9(6.89)
Quality of Filming	1	2	3	4	5	6	7▲	8	9(7.22)
Quality of Acting	1	2	3	4	5▲	6	7	8	9(5.67)
Quality of Feedback	1	2	3	4	5	6	7▲	8	9(7.22)

12. To what extent did you agree with the content of the training for the three approaches?

	Agreed with none of the Training								Agreed with all of the Training	
Role Playing	1	2	3	4	5	6	7▲	8	9(7.54)	*
Textbook	1	2	3	4	5▲	6	7	8	n.s. 9(5.62)	*
Videodisc	1	2	3	4	5	6	7▲	8	9(7.20)	

13. Do you have any comments or suggestions for leadership training or the way this research was conducted?

Table 42

Summary Means for Leadership Principles Test (Raw Scores)

```
ZANOVA

FILE   VISTASYS
SUBFILE   V1        V2        V3        V4        V5        V6

* * * * * * * * * * * * * * * * C E L L   M E A N S * * * * * * * * * * *
              SCORE
           BY VISTA
              MODE
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *


TOTAL POPULATION

    11.35
 (   312)


                    (Evaluation Replication #)
VISTA
        1         2         3         4         5         6

    10.40     9.85     10.93     14.67     12.02     11.49
 (    90)  (    54)  (    38)  (    44)  (    45)  (    41)


MODE (Videodisc)   (Text)    (Role Play)
        1          2          3

    12.37      10.51      11.18
 (   103)   (   104)   (   105)


             MODE
                    1          2          3
VISTA
        1       11.38      10.33       9.33
             (    30)   (    29)   (    31)

        2       11.61       8.74       9.06
             (    19)   (    18)   (    17)

        3       10.52      10.36      11.79
             (    11)   (    11)   (    14)

        4       15.71      11.78      16.64
             (    15)   (    15)   (    14)

        5       13.00      10.89      10.48
             (    14)   (    15)   (    14)

        6       10.79      11.50      12.23
             (    14)   (    14)   (    13)
```

Table 43

ANOVA Results for Overall Leadership Principles Test (Raw Scores)

```
ZANOVA

FILE   VISTASYS
SUBFILE   V1          V2          V3          V4          V5          V6

* * * * * * * * * * A N A L Y S I S   O F   V A R I A N C E * * * * * * * * * *
                SCORE
             BY VISTA
                MODE
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
```

SOURCE OF VARIATION	SUM OF SQUARES	DF	MEAN SQUARE	F	SIGNIF OF F
MAIN EFFECTS	902.115	7	128.874	7.316	0.000
VISTA	717.571	5	143.514	8.147	0.000
MODE	188.197	2	94.099	5.342	0.005
2-WAY INTERACTIONS	364.051	10	36.405	2.067	0.027
VISTA MODE	364.050	10	36.405	2.067	0.027
EXPLAINED	1266.164	17	74.480	4.228	0.000
RESIDUAL	5179.168	294	17.616		
TOTAL	6445.332	311	20.725		

```
312 CASES WERE PROCESSED.
  0 CASES (  0.0 PCT) WERE MISSING.
```

Table 44

Summary Means for Leadership Principles Test (T Scores)

```
ZANOVA

FILE   VISTASYS
SUBFILE   V1          V2          V3          V4          V5          V6

* * * * * * * * * * * * C E L L   M E A N S * * * * * * * * * *
              T
          BY VISTA
             MODE
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

TOTAL POPULATION

    50.00
 (  312)

                                (Evaluation Replication #)
VISTA
        1           2           3           4           5           6

    50.00       50.00       50.00       50.00       50.00       50.00
 (   90)  (   54)  (   38)  (   44)  (   45)  (   41)

MODE(Videodisc)   (Text)      (Role Play)
        1           2           3

    52.30       48.27       49.46
 (  103)  (  104)  (  105)


          MODE
                    1           2           3
VISTA
        1       52.81       49.80       47.47
             (   30)  (   29)  (   31)

        2       54.40       47.22       48.02
             (   19)  (   18)  (   17)

        3       48.95       48.56       52.17
             (   11)  (   13)  (   14)

        4       51.75       45.15       53.32
             (   15)  (   15)  (   14)

        5       55.77       47.80       47.01
             (   14)  (   15)  (   16)

        6       48.09       50.03       52.02
             (   14)  (   14)  (   13)
```

Table 45

ANOVA Results for Overall Leadership Principles Test (T Scores)

```
ZANOVA

FILE   VISTASV5
SUBFILE   V1          V2          V3          V4          V5          V6

* * * * * * * * * * A N A L Y S I S   O F   V A R I A N C E * * * * * * * * * *

                T
            BY VISTA
               MODE
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
```

SOURCE OF VARIATION	SUM OF SQUARES	DF	MEAN SQUARE	F	SIGNIF OF F
MAIN EFFECTS	887.031	7	126.719	1.324	0.239
VISTA	1.027	5	0.205	0.002	1.000
MODE	887.031	2	443.516	4.633	0.010
2-WAY INTERACTIONS	1566.115	10	156.612	1.636	0.096
VISTA MODE	1566.115	10	156.612	1.636	0.096
EXPLAINED	2453.148	17	144.303	1.507	0.091
RESIDUAL	28146.332	294	95.736		
TOTAL	30599.480	311	98.391		

```
    312 CASES WERE PROCESSED.
      0 CASES (  0.0 PCT) WERE MISSING.
```

Table 46

Overall Summary (Mean Ratings)

Name _____

Service # _____

Group _____

S# _____

Note: * denotes $p < .05$
n.s. denotes $p > .05$

IOBC

LEADERSHIP TRAINING

PREFERENCE

1. How much do you think you would learn about dealing with people using the following approaches: (Circle one in each row)

	Nothing 1	2	3	4	5	6	7	8	9 A great deal		
Role Playing:							▲			7.08	*
Textbook:					▲					4.69	*
Videodisc:							▲			6.60	*

2. How useful are the follcwing three types of leadership training to Second Lieutenants like yourself? (Circle one in each row)

	Useless 1	2	3	4	5	6	7	8	9 Extremely useful		
Videodisc:							▲			6.77	*
Textbook:					▲					4.91	*
Role Playing:							▲			7.33	*

3. How did the training keep your attention? (Circle one in each row)

	Very boring. I lost interest. 1	2	3	4	5	6	7	8	9 It was very interesting.		
Textbook:					▲					4.44	*
Videodisc:							▲			7.24	n.s.
Role Playing:							▲			7.41	*

Overall Results (Continued)

4. How motivated did the training methods keep you?
(Circle one in each row)

	It didn't motivate me.							It was very motivating.					
Textbook:	1	2	3	▲4	5	6	7	8	9	3.94	*		
Role Playing:	1	2	3	4	5	6	▲7	8	9	7.04		*	
Videodisc:	1	2	3	4	5	6	▲7	8	9	6.80			n.s.

5. In terms of learning about Army leadership, I think that Role Play (172) is the best training approach, Videodisc (109) is second best, and Text (15) is third.
(Fill in blanks with Textbook, Role Playing, and Videodisc.)

6. In terms of keeping my interest, I think that Role Play (159) is the best training approach, Videodisc (132) is second best and Text (5) is third.
(Fill in the blanks with Textbook, Role Playing, and Videodisc.)

7. Would you prefer to see a mixture of the three approaches in the Counseling Laboratory?

YES	NO
90.5%	9.5%

8. If yes, what %?

	17.6%	Textbook
+	36.8%	Videodisc
+	45.5%	Role Playing
	100%	

In the Role Playing session did you (check one): 32.2% Role Play
67.8% or just watch

9. When you were shown the videodisc approach, you were shown two teaching techniques. The Pedagogical approach provided text feedback about why answers were the best or not the best and the Experiential allowed events go on without too much interruption (simulated a real interaction).

Which of these two do you think is more valuable in learning leadership?

29.8% Experiential
or
70.2% Pedagogical

(Note: only Videodisc subjects)

10. Do you think they should be combined? 67.4% Yes or 32.6% No

If "Yes" how should they be combined?

11. Evaluate the following for the videodisc:

	Distracted from Training								Contributed to Training (Medians)
	1	2	3	4	5	6	7	8	
Quality of Writing							▲		9 (6.86)
Quality of Filming								▲	9 (7.34)
Quality of Acting							▲		9 (6.60)
Quality of Feedback								▲	9 (7.38)

12. To what extent did you agree with the content of the training for the three approaches?

	Agreed with none of the Training								Agreed with all of the Training	
	1	2	3	4	5	6	7	8		
Role Playing							▲		9 (7.29)	
Textbook						▲			9 (5.92)	n.s.
Videodisc							▲		9 (7.24)	*

13. Do you have any comments or suggestions for leadership training or the way this research was conducted?